Sitecore Experience Platform 8.0

All the official Sitecore documentation.

Experience Analytics glossary

This topic contains a glossary of Sitecore Experience Analytics terms, with short definitions.

Asset

A piece of content, for example, a PDF file, images, videos, or downloadable interactive content, that you can use to attract contacts to your website and increase their engagement with your organization.

Average Duration

The average duration of the visit.

Branded Keyword

A keyword typically used in a search engine that contains a company's name or brand, for example, Sitecore Experience Platform.

Bounce

A visit from a contact who leaves after only viewing one page.

Bounce Rate

The percentage of contacts who enter your website and leave after only viewing one page.

Campaign Facet

A customizable taxonomy tag that allows you to classify your campaigns into meaningful groups, for example, by geographic region or business unit, and that enables you to track how contacts interact with different types of campaign activities.

Conversion

An action that shows a measurable commitment between a contact and an organization occurring when a contact completes a goal.

Conversion Rate

The percentage of visitors that achieve a conversion. A conversion rate is calculated by dividing the number of conversions by the number of visits, and multiplying the result by 100.

Count per Visit

The number of times an outcome occurs in a visit. (For example, how many goals or outcomes are triggered per visit.)

Dimension

Something that you can measure about an interaction. (For example, you can measure how many goals were triggered in an interaction.)

Engagement Value

The accumulated sum of engagement value points generated by a contact during one or more visits and used in Experience Analytics to determine the degree of commitment exhibited by a contact on a website.

Entry page

Any page on a website that is a contact’s entry point in a visit.

Exit page

The last page accessed during a visit, or in a multiwindow browser, that indicates the end of a visit.

Fact

A specific recorded measure for a dimension. (For example, between 10 am and 11 am the Register goal was triggered ten times.)

Interaction

The exchange of communication between a contact and a brand that takes place through one or more channels.

Monetary Value

The amount of money associated with an outcome.

Monetary Value per Visit

The average monetary value per visit.

Non-organic keyword

A search term that comes from a paid search function.

Non-branded keyword

A search term that does not contain a company's name or brand within it, for example, content management system.

Organic Keyword

A search term that does not come from a paid search function.

Outcome

A business-significant result of one or more interactions between a contact and your organization, for example, a sales lead or product purchase, and that you can use to quantify your contacts based on the financial value they bring to your organization.

Page Views per Visit

The average number of website pages a visitor views per visit.

Pattern Card

A predefined set of profile values used to define personalization rules for particular types of contacts or segments.

Referring site

An external website that contains a link to the current website.

Taxonomy

A classification system that you use to organize marketing activities by applying taxonomy tags to campaigns, goals, and events in the Marketing Control Panel.

Time on Page

The length of time that a contact spent on a given page.

Value per Visit

The average engagement value generated per visit. This value can be used to determine the effectiveness of a campaign.

Visit

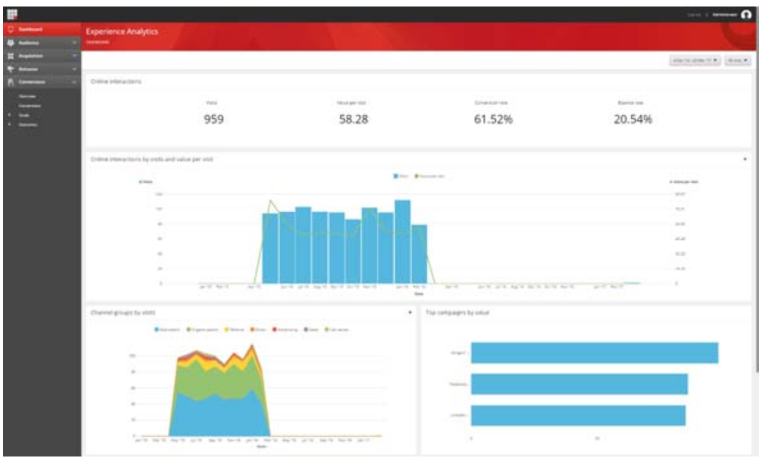
All the actions that a visitor takes from the moment they enter a website until the moment they leave and that consists of all the pages viewed, and resources consumed, such as campaigns triggered and conversions achieved.

Send feedback about the documentation to docsite@sitecore.net.

Experience Analytics overview

Sitecore Experience Analytics provides dashboards and reports for marketers and marketing analysts to identify patterns and trends in experience data collected from their websites and potentially other external data sources.

A sample Experience Analytics report:



When you open Experience Analytics for the first time, you see a dashboard page, which provides an overview of your experience data. There are several categories of reports, each containing different charts and graphs.

You can use experience data to analyze the performance, optimization, and the level of commitment of your website visitors and contacts by comparing metrics such as page views, conversion rates, and engagement value.

Marketers can view a dashboard, and multiple reports. They can also filter report data by date range or by website.

Note

Although Sitecore provides sample marketing definition items, when you start a marketing plan or campaign, you must create your own taxonomy of marketing definition items.

After you create the definition items, you must deploy them and then create a custom report to help you interpret the data.

To perform more advanced tasks, developers can use Visual Studio and Sitecore Rocks to create new reports, edit, and add charts and graphs or create advanced filters.

Experience Analytics contains the following categories of reports:

- *Dashboard* – provides an overview of key analytics by displaying a selection of charts and performance indicators.
- *Audience* – helps you gain an understanding of who your visitors are.
- *Acquisition* – shows you what is driving traffic to your website. For example, from campaigns and other marketing channels.
- *Behavior* – helps you to analyze the behavior of your visitors. For example, by analyzing the effectiveness of your profiling and personalization strategy.
- *Conversions* – the percentage of visitors that achieve a particular goal. This helps you understand how well your marketing efforts are working.

Important

If you want to change or customize a standard report that comes with Experience Analytics, it is best practice to copy an existing report, customize it and then rename it.

This is due to possible changes in standard reports in future versions of Sitecore.

Send feedback about the documentation to docsite@sitecore.net.

Change the metrics in an existing analytics report

In Experience Analytics, you can create a custom analytics report from scratch with custom metrics. However, in existing reports, you can also change the default metrics if they do not suit you.

In Sitecore, you use metrics in combination with dimensions to create reports. Metrics are a value, measurement, or a count of a characteristic tracked for a dimension. They can further break down a report based on common criteria, for example, *Conversions* or *Bounces*.

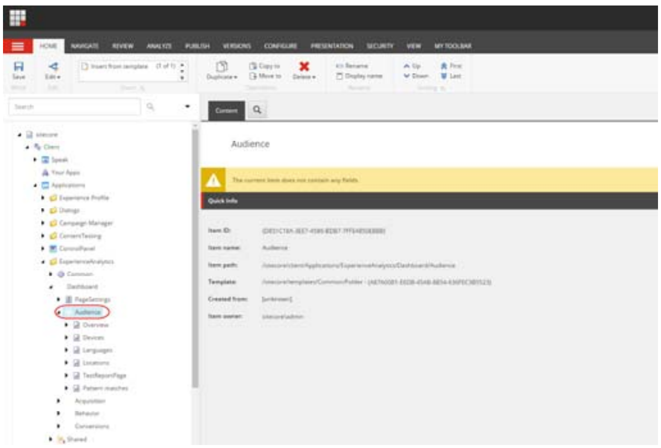
Dimensions group the data used in reports, for example, in an interaction or campaigns triggered. Experience Analytics provides several default metrics and dimensions.

Important

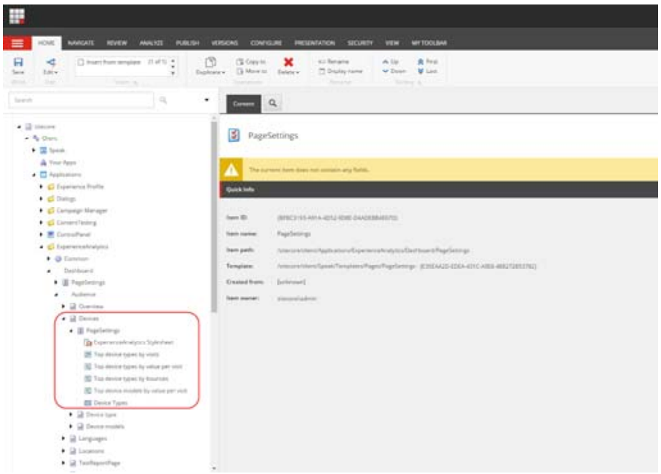
If you want to change or customize a standard report that comes with Experience Analytics, it is best practice to copy an existing report, customize it, and then rename it. This is due to possible changes in standard reports in future versions of Sitecore.

To change the metrics in an existing report:

1. In the Core database, in the Content Editor, navigate to and expand the *Dashboard* item (*/sitecore/Client/Applications/ExperienceAnalytics/Dashboard*), and then expand the subitem where the report you want to customize is located. For example, on the Audience tab:



2. Expand the subitem where your report is located, and expand its *PageSettings* item. For example, *Devices*, *PageSettings*.



3. Click the report you want to customize. For example, *Top device types by visits*.
4. In the right pane, in the Data section, in the Metric field, click the drop-down arrow and select the metric you want to see displayed in the report. For example, *Value per visit*.
5. Click Save to save your changes.

Note

After you change the default metrics in a report, it is best practice that you change the name of the report to reflect the change in the metrics.

Send feedback about the documentation to docsite@sitecore.net.

Create a custom report filter and segment

In Experience Analytics, you use dimensions and segments to create charts used in reports.

Dimensions and segments work together to define the data available in the reports. Dimensions determine how visits are analyzed, segments control which visits are analyzed. For example, you could have a dimension called *By Goal* and then a segment below it called *All visits by Goal*.

Experience Analytics comes with several default dimensions, and segments, but you can also add your own if your business requires it. You can create segments by combining a dimension with a filter.

You can also create new segments in Sitecore but creating new dimensions is a developer task.

To create a custom report filter and segment:

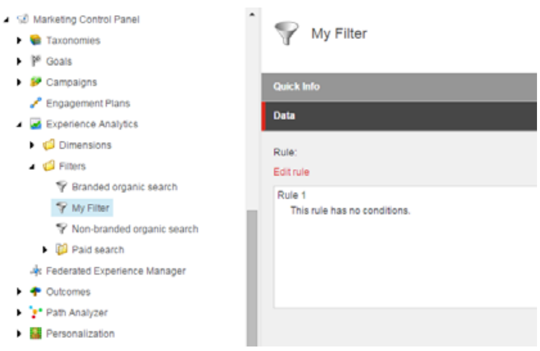
- Create a custom filter
- Create a custom segment and link it to your filter
- Add a custom segment to a report

Create a custom filter

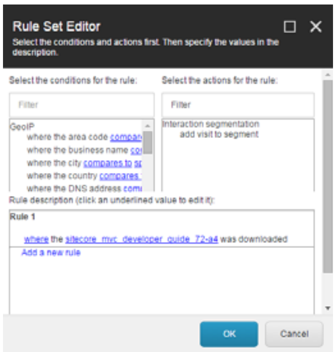
In the Content Editor, you can create a filter to use with segments.

To create a new filter item:

1. In the Content Editor, navigate to */sitecore/system/Marketing Control Panel/Experience Analytics/Filters*.
2. In the Sitecore ribbon, on the Home tab, in the Insert group, click Filter.
3. In the Message dialog box, enter a name for your filter, for example, *My Filter* and click OK.
4. Click Save. To configure your new filter item, in the right pane, in the Data section, click Edit Rule.



5. To create a rule for your filter, in the *Rule Set Editor*, select a condition. For example, you can create a filter for a report that displays statistics on a contact that has downloaded a brochure from your website. To do this, in the Select the conditions for the rule field, search for *Asset,* and then select the rule *where the Asset was downloaded*.



6. To select the asset, in the Rule description section, in the Rule 1 field, click the underlined word Asset.
7. In the Select Item dialog box, navigate to the folder where your asset is located, and select the asset. Click OK.
8. In the right pane, in the interaction segmentation section, click the *add visit to segment* action.
9. Click OK to save the rule.

Create a custom segment and link it to your filter

The next step is to create a segment item to link your new filter to.

To create a segment for your filter:

1. In the Content Editor, in the content tree, in the *Dimensions* item (*/sitecore/system/Marketing Control Panel/Experience Analytics/Dimensions*), click the *Email*, *Pages* or *Visits* folder where dimensions are located.
2. Click the appropriate dimension to add your custom segment to, for example, the By Downloads dimension.
3. In the Sitecore ribbon, on the Home tab, in the Insert group, click Segment to create a custom segment item, and in the Message dialog box enter a name, for example, *My Segment* and then click OK.
4. To connect your custom segment to your custom filter, in the right pane, in the Data section, in the Filter field, click the drop down arrow and select the filter that you created earlier.

5. In the Sitecore ribbon, in the Review tab, in the Workflow group, click Deploy to save your changes and start data collection for this segment. By default, data collection starts 30 minutes after deployment.

Note

Once you have deployed a new segment and data collection has begun you cannot change the segment as this could cause inconsistencies in your collected data.

Add a custom segment to a report

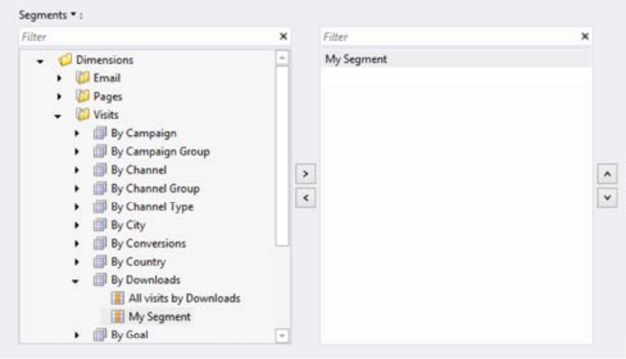
In Sitecore Rocks, add your new custom segment to an existing report:

1. In *Sitecore Rocks,* navigate to the report that you want to edit. For example, navigate to:
2. `<sitename>core/sitecore/client/Applications/ExperienceAnalytics/Dashboard/Behavior/Downloads/PageSettings/Downloads`
3. Select a line chart control to edit. To do this for the Download report, right click Asset, click Tasks then click Design Layout.

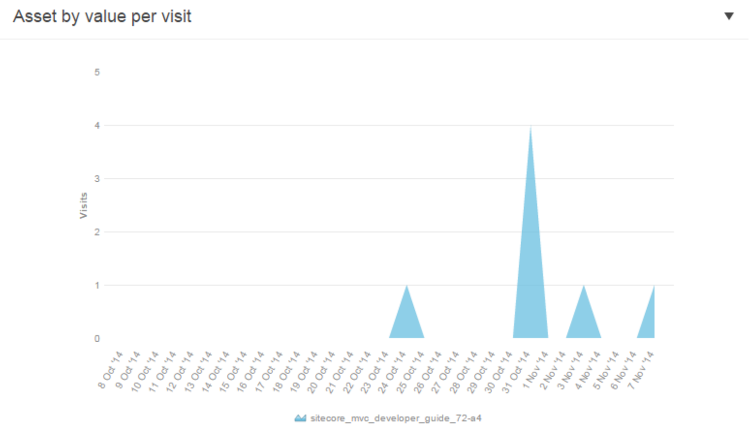
ColumnPanel 3.1	ColumnPanel	RowPanel 3.Content
ColumnPanel 3.2	ColumnPanel	RowPanel 3.Content
AssetCountAndValuePerVisit	ExperienceAnalyticsLineChart	ColumnPanel 1.1.Content
AssetByConversionRate	ExperienceAnalyticsAreaChart	ColumnPanel 2.1.Content
ExperienceAnalyticsAreaChart2	ExperienceAnalyticsAreaChart	ColumnPanel 2.2.Content
ExperienceAnalyticsAreaChart3	ExperienceAnalyticsAreaChart	ColumnPanel 3.1.Content
ExperienceAnalyticsAreaChart4	ExperienceAnalyticsAreaChart	ColumnPanel 3.2.Content

4. To edit a line chart control, right click the relevant line chart and then click Edit Data Source, for example, the *AssetCountAndValuePerVisit* line chart.

   In the Data group, you can see the new custom segment that you created under Segments in the Dimensions folder.



5. Select the custom filter and then click the right arrow button to make the custom segment visible in a chart.
6. In *Visual Studio*, *Sitecore Rocks*, save your changes.
7. In Experience Analytics, you can view the changes to the report.



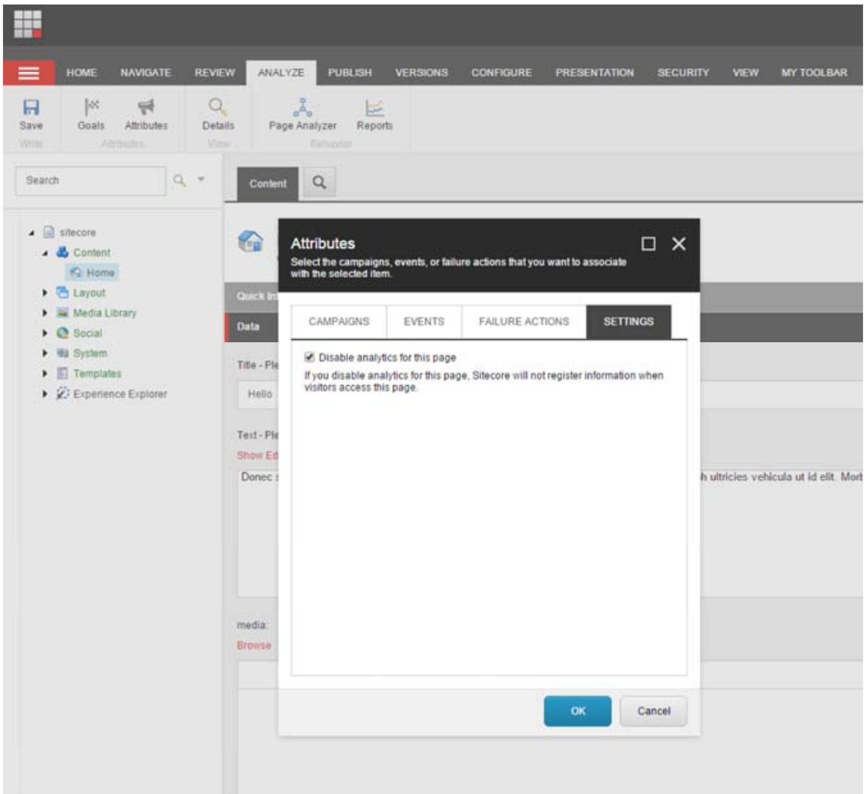
Send feedback about the documentation to docsite@sitecore.net.

Disable Analytics on an item

The Analytics attribute enables you to view data on user activity for a particular item in Experience Analytics. Analytics is enabled by default on your items, but you can turn off the Analytics setting in the Content Editor. This is useful if you, for example, have an item that automatically redirects visitors to another item. You may not want such an item to appear in reports and session trails.

To disable or enable analytics on an item:

1. In the Content Editor, expand the content tree and navigate to the content item that you want to change the analytics setting for.
2. On the ribbon, on the Analyze tab, in the Attributes group, click Attributes.



3. In the Attributes dialog, click the Settings tab and select or clear the Disable Analytics for this Item check box to assign this setting to the content item.

   When you select Disable Analytics for this Item, the item is no longer visible in the visitor session trail or any other reports in Sitecore.

4. Click OK to save your changes and close the Attributes dialog.

Send feedback about the documentation to docsite@sitecore.net.

Experience Analytics advanced report filtering

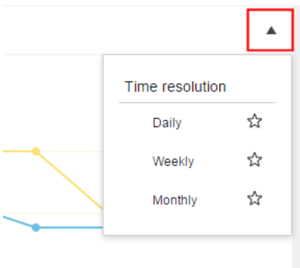
In Experience Analytics you can view a selection of dashboards and reports that enable you to get an overview of the patterns and trends in your experience data.

You can perform more advanced tasks using Visual Studio and Sitecore Rocks, for example to:

- Change the default time resolution in a report
- Change the keys displayed in a report

Change the default time resolution in a report

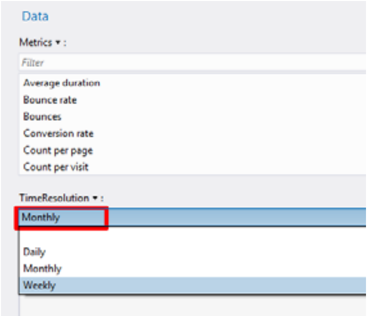
In Experience Analytics, when you open a report, click the drop-down icon to the right of the current report to see the available time resolution options.



The default time resolution setting is automatically chosen depending on the date range when viewing the report. You can change this setting so that another value is the default, however, it is considered best practice if you leave the `TimeResolution` field blank.

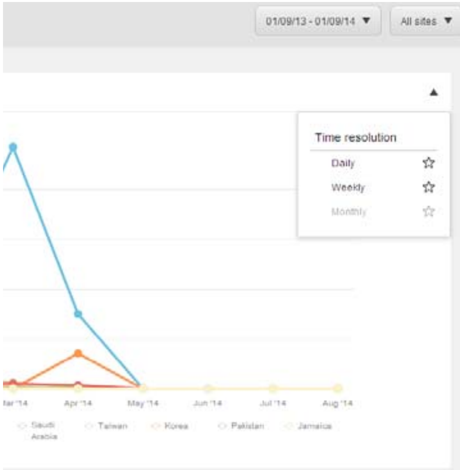
To change the default time resolution in a report:

1. In Visual Studio, in Sitecore Rocks, open a report, for example, *TestReportPage*. Right click the report item, click Tasks and then click Design Layout.
2. In the report design layout, right click the ExperienceAnalyticsLineChart item and click Edit Data Source.
3. In the parameters item, under Data, TimeResolution, click a value in the drop-down list, for example, Monthly.



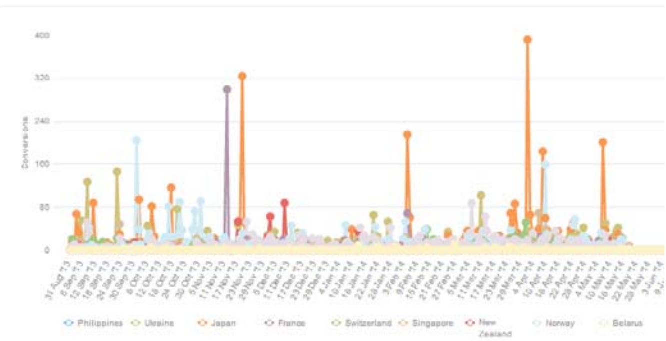
4. Save your changes.

   In Experience Analytics, you can view the *TestReportPage* report. The grayed out value is the default or currently selected time resolution value.



Change the keys displayed in a report

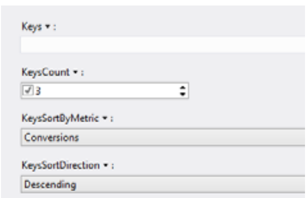
In Experience Analytics, when you open a report there are several metrics, keys or segments displayed along each of its axis. In the following example, the report shows conversions by country. Each country is represented as a different key.



This report displays all the country keys. A developer can change this filter so that only specific keys are displayed.

To change the keys displayed in a report:

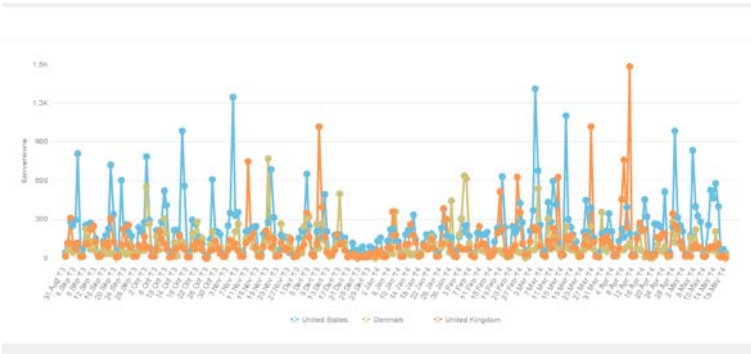
1. In Visual Studio, in Sitecore Rocks, open a report, for example, *TestReportPage*. Right click the report item, click Tasks and then click Design Layout.
2. In the report design layout, right click the ExperienceAnalyticsLineChart item and click Edit Data Source.
3. Expand the Data group, select the KeysCount check box, and enter *3* in the KeysCount field:



4. Save your changes.

   In this example, the report only displays the top three countries for conversions and sorts them in descending order.

   You can test your filter in Experience Analytics by viewing the report that you have modified.



You can use the same filter on all of the following types of controls:

- *ExperienceAnalyticsBarChart*
- *ExperienceAnalyticsAreaChart*
- *ExperienceAnalyticsLineChart*
- *ExperienceAnalyticsListControl*

Send feedback about the documentation to docsite@sitecore.net.

Import historical data into Experience Analytics

In a live Sitecore Experience Platform installation, the xDB is optimized to process a continuous, steady stream of contact interactions for use in Experience Analytics reports.

When you upgrade from an existing installation to a newer version of Sitecore, you may have a significant number of historical interactions that you want to see in Experience Analytics after upgrading. In this situation, the process of extracting all historical data can be very resource intensive and have a negative effect on performance.

By default, Experience Analytics ignores historical data and only processes new interactions. New interactions are defined as interactions saved after the deploy date of a segment.

Changing the deploy date to a date in the past means that your report segments can now also pick up historical data if you rebuild your reporting database.

Note

The more historical data that you want to process, the longer it takes to rebuild the reporting database.

Therefore, setting a deploy date requires careful thought and planning and you may decide to only include one year of historical data rather than several to avoid having a negative effect on performance.

To import historical data into Experience Analytics:

1. Run the following SQL Server query on your primary reporting database to specify the segment deploy date that you want to use for live processing. In the following example, the xDB contains data going back to 2008 but you only want to include data from 2010 in your Experience Analytics reports, so set the deploy date to 2010-01-01 as follows:

   ```
   UPDATE [Sitecore8.Analytics.primary].[dbo].Segments SET DeployDate='2010-01-01 00:00:00.000'
   ```

2. Clear the website cache. To clear the cache, run the `/Sitecore/admin/cache.aspx` page in a web browser and click Clear All.
3. Open Microsoft SQL Server Management Studio and run the following query on the secondary reporting database:

   ```
   SELECT * FROM [Sitecore8.Analytics.secondary].[dbo].Segments
   ```

   It is expected that this query returns no results.

4. In SQL Server, in the same query window, enter the following SQL UPDATE statement on the secondary reporting database. For example, set the deploy date to `2010-01-01` but do not run the query yet:

   ```
   UPDATE [Sitecore8.Analytics.secondary].[dbo].Segments SET DeployDate='2010-01-01 00:00:00.000'.
   ```

5. Start the rebuild process by running the `/Sitecore/admin/rebuildreportingdb.aspx` page in a web browser.
6. When the Process State reaches *WaitReadyToReceiveData*, immediately run the SQL UPDATE statement that you prepared earlier in step 4. This state is displayed directly after the *StoppingDataRecording* process state.

   Important

   You must execute the SQL Server SQL UPDATE statement before `rebuildreportingdb.aspx` moves on to the *HistoryProcessing* process state.

Send feedback about the documentation to docsite@sitecore.net.

Set a report segment initialization time period

In Experience Analytics, when you deploy a new report segment, an initialization time period (padding) is automatically set in the `Sitecore.ExperienceAnalytics.Aggregation.config` file.

An initialization period is necessary to allow adequate time for the synchronization of all the dependent processes and servers in your Sitecore environment. During live aggregation, only interactions saved after the initialization period of a segment will be processed by that segment. If the initialization period is too short, you may get inconsistent data until the synchronization process has completed.

The `SegmentDeployDateOffsetPadding` setting enables you to set an initialization time period. After you have configured this setting, any new segments you create will use the updated value when you deploy them.

The next time you run live aggregation processing, any new interactions added to the xDB collection database will not be displayed in the new segment until the initialization period has expired.

The default initialization time period setting is 30 minutes but you can change this depending on your requirements.

Important

You should only change the time value setting after very careful consideration. It is important to be aware that entering a lower value could make xDB data around your deploy date inconsistent. You should only increase this value if your processing environment is part of a large distributed environment with high latency.

Note

Do not make changes directly to the configuration files. Instead, you must create a patch file that performs the required changes during runtime.

To configure the initialization time period setting:

1. In the `Sitecore.ExperienceAnalytics.Aggregation.config` file (`App_Config\Include\ExperienceAnalytics`) navigate to the following setting:

   ```
   <setting  name="ExperienceAnalytics.Aggregation.SegmentDeployDateOffsetPadding" value="00:30:00" />
   ```

2. Enter a time value in this parameter to set your required initialization period.
3. Save your changes.

Send feedback about the documentation to docsite@sitecore.net.

Set up branded keywords

In Sitecore Experience Analytics, you can set up a list of predefined branded keywords, so you can separate them from non-branded keywords.

A branded keyword includes your brand or company-specific name, while a non-branded keyword does not. By separating branded and non-branded keywords, you can improve your marketing effectiveness and customer engagement, as well as measure brand awareness.

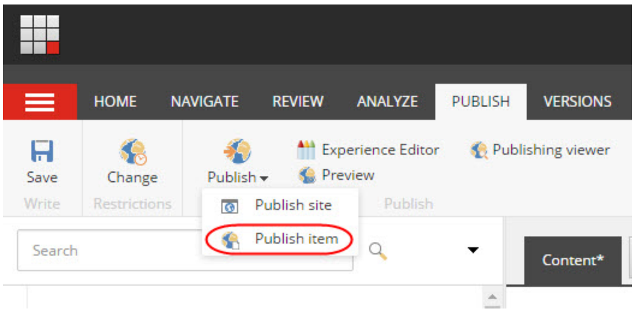
For example, if you add the keywords *Sitecore*, *Sitecore CMS,* and *Experience Platform* to the list, when a contact searches for any of those terms in a search engine, and then clicks on a link to come to http://www.sitecore.net, Experience Analytics identifies the keyword used by the contact as branded.

You can add as many branded keywords to the list as you like.

To set up your branded keywords:

1. In the Content Editor, in the content tree, navigate to and select the *Organic Branded Keywords* item *(sitecore/System/Settings/Analytics/Organic Branded Keywords).*
2. In the right pane, in the Data section, in the Keywords field, enter the keywords that identify a branded search, separated by a line break. For example:

3. Click Save.
4. On the ribbon, on the Publish tab, in the Publish group, click the Publish drop-down arrow and click Publish item.



In future searches, when a contact enters one of the predefined keywords from the list in a search engine, and then clicks a link, the keyword appears in the Experience Analytics dashboard. You can see it on the Acquisition tab, on the Keywords subtab, in the *Organic branded* report.

Send feedback about the documentation to docsite@sitecore.net.

Walkthrough: Creating an Experience Analytics report

Sitecore Experience Analytics provides dashboards and reports for marketers and marketing analysts to identify patterns and trends in experience data. This data is collected from websites and other external data sources.

You can create an experience analytics report from scratch using Microsoft Visual Studio and Sitecore Rocks.

This walkthrough outlines how to:

- Create a new report page item
- Add a chart component to a report
- Add a parameters item to the report
- Bind the bar chart control to the parameters item
- Change the chart title
- Test the report

Important

If you want to change or customize a standard report that comes with Experience Analytics, it is best practice to copy an existing report, customize it, and then rename it. This is due to possible changes in standard reports in future versions of Sitecore.

Create a new report page item

Add a new report page to the content tree. A report page contains other subitems that you need when you configure your report.

Note

You need to install Microsoft Visual Studio with Sitecore Rocks, and a suitable Sitecore website to complete this task.

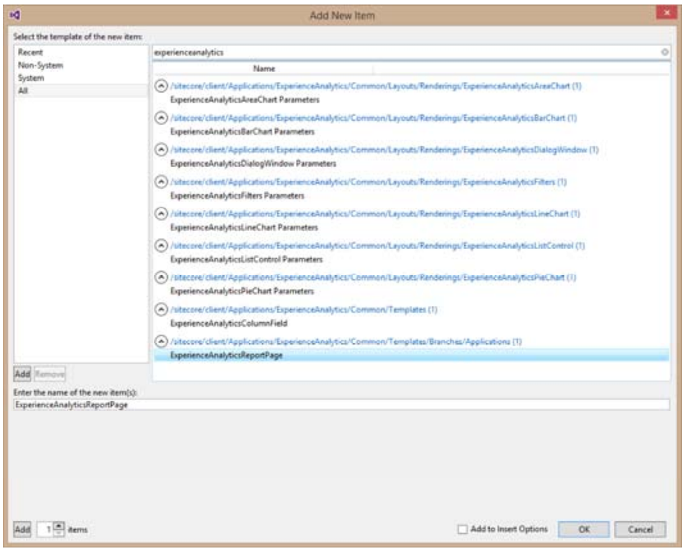
To create a report page:

1. Open Sitecore Rocks and create a connection to your website.
2. In the website content tree, navigate to the Dashboard node using the following path:

   `core/sitecore/client/Applications/ExperienceAnalytics/Dashboard`

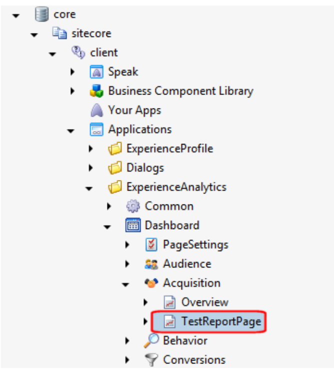
3. To add a new report page under this item, select a report category, for example, *Acquisition*. Right click the category, click Add, and then click New Item.
4. In the New Item dialog box, click *All* to see all available SPEAK report templates.

   To find all available report templates, enter a search term beginning with *experienceanalytics.*



5. Select the *ExperienceAnalyticsReportPage.* You can accept the default name or provide a name for the item, for example, *TestReportPage*.
6. Click OK to save your changes.

   Your new report page appears in the content tree below the category, for example, *Acquisition*, and it also contains a *PageSettings* subitem.



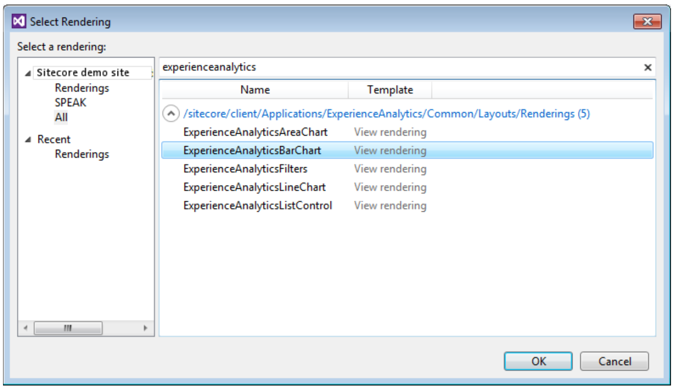
Add a chart component to a report

Before you can add a chart component to a report, you first need a report page.

To add a chart component to a report page:

1. In Sitecore Rocks, right-click a report page, click Tasks and then click Design Layout.
2. In the Design Layout window, click Add Rendering.
3. In the Select Rendering dialog box, select a chart control to use in your report page.

   To find all available chart controls, enter a search term beginning with *experienceanalytics*. You can choose from an area chart, bar chart, list, or line chart. For example, to display a bar chart, select the *ExperienceAnalyticsBarChart* rendering.

4. Choose a chart type and click OK to save your changes.

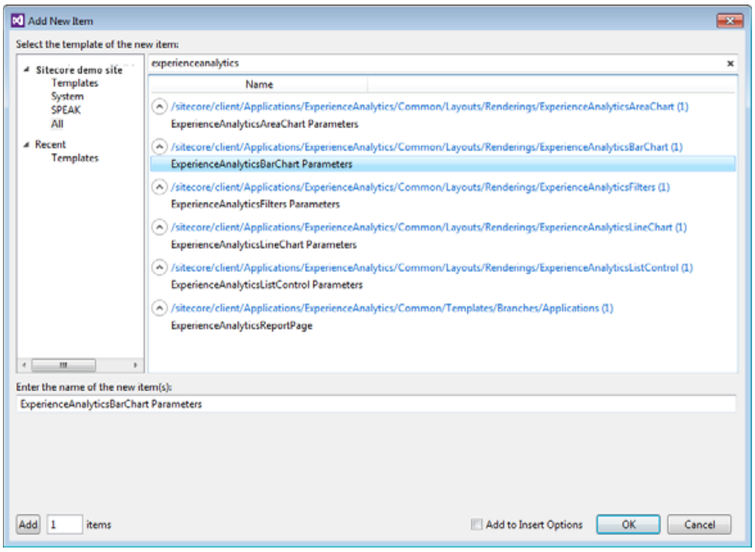
Add a parameters item to the report

You need a parameters item to configure report metrics and other settings in your chart. A parameters item is a subitem of the `PageSettings` item.

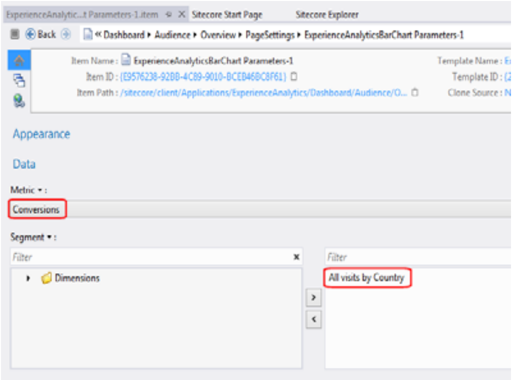
To add a parameters item:

1. On the report page, right-click the PageSettings item and click Add - New Item.

   To find all available chart parameter controls, enter a search term beginning with *experienceanalytics*. You can see parameters items for each type of chart control. For example, if you are using the bar chart control, select the *ExperienceAnalyticsBarChart Parameters* item.

2. You can enter a name for your parameters item or accept the default name, then click OK.
3. To configure the parameters item, double click the item to open it.
4. In the parameters item, select a metric and add a segment to your chart.



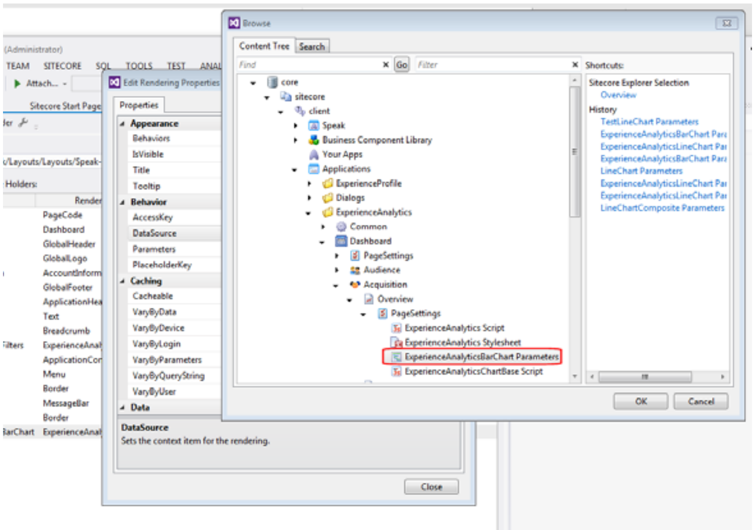
5. For example, to show conversions from all visits by country, select the following:
   - Metric = *Conversions*
   - Segment = *All visits by Country*
6. Save your changes.

Bind the bar chart control to the parameters item

If you do not bind the bar chart control to the parameters item, your report will not display any data.

To bind the *ExperienceAnalyticsBarChart* control to the parameters item:

1. Right-click the report, click Task, Design Layout, and then double-click the chart item that you have created. For example, double click the *ExperienceAnalyticsBarChart* control.
2. In the Edit Rendering Properties dialog box, in the Behavior section, in Data Source, browse to the location of the *ExperienceAnalyticsBarChart Parameters* item in the content tree.



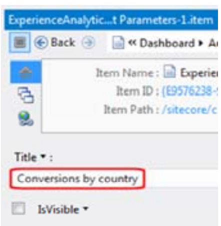
3. Select the parameters item and click OK.
4. Save your changes.

Change the chart title

Your new chart needs a heading to describe the content that it displays.

To change a title label on a chart:

1. Double-click the parameters item for your chart.
2. In the parameters item, expand the Appearance section.
3. In the Title field, enter the relevant title, for example, *Conversions by country*.



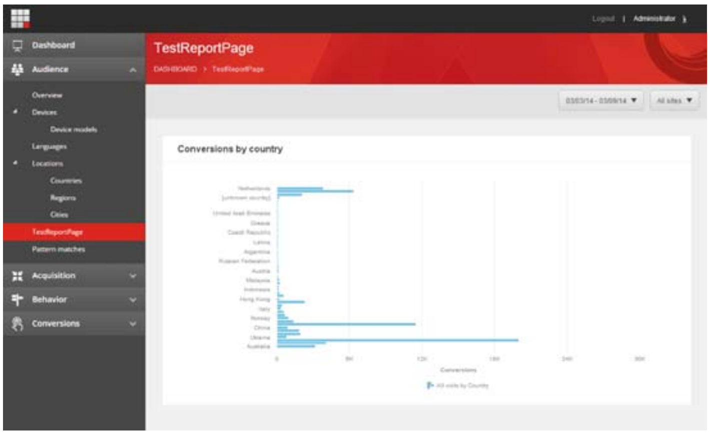
4. Save your changes and view the chart in Experience Analytics.

Test the report

When you have successfully created your new report, open *Experience Analytics* and preview the report in a web browser.

To test the report:

1. Open the Experience Analytics dashboard.
2. Navigate to the relevant tab and select your new report.



3. If you do not see anything in your report, select a wider date range of data and all sites.

Send feedback about the documentation to docsite@sitecore.net.

Filter an Experience Analytics report

In Experience Analytics, marketers and marketing optimization analysts can view a selection of dashboards and reports to gain a quick understanding of patterns and trends in experience analytics data.

You can also apply basic filters, such as by date range or website, as well as show or hide lines and bars in a chart or graph.

This topic outlines the following procedures:

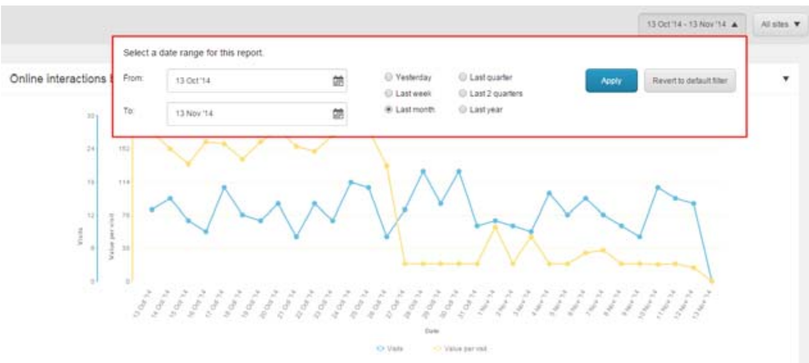
- Filter a report by date range
- Filter a report by website
- Show or hide lines and bars in a line chart or graph

Filter a report by date range

To see more data in a report you can use a filter to select a wider date range.

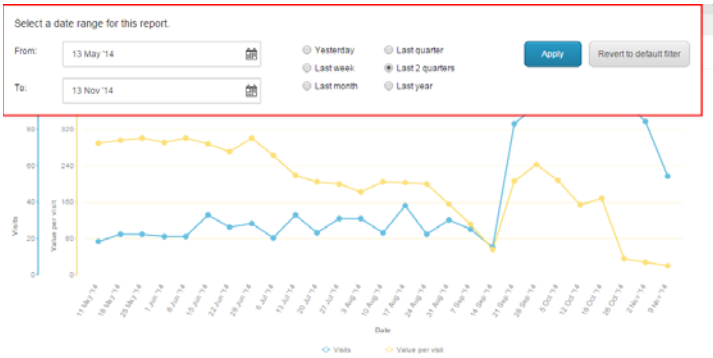
To filter an Experience Analytics report by date range:

1. In the Experience Analytics dashboard, open a report, for example, *Audience Overview*.
2. Click the currently selected date range to open the date range filter dialog box:



3. By default, you see one month of data. Use the calendar to select a *From* date and a *To* date, or select a predefined time range using the radio buttons. For example, *Last 2 quarters*.
4. Click Apply.

   After applying the filter the report displays more data:



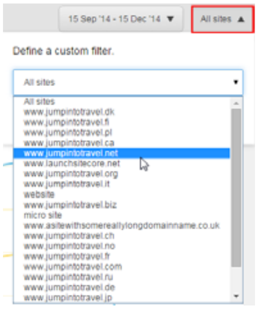
5. To clear the filter, click *Revert to default filter*.

Filter a report by website

Use the website filter list to view data for one website or all sites at the same time.

To filter a report by website:

1. In the Experience Analytics dashboard, open the report that you want to filter, for example, *Audience Overview*.
2. To open the website filter list, click All sites.



You can view data from all your websites simultaneously or select a single website.

3. Select the website, for example, *www.jumpintotravel.net,* and click Apply.

   Now you can only see reports containing experience data from the *www.jumpintotravel.net* website.

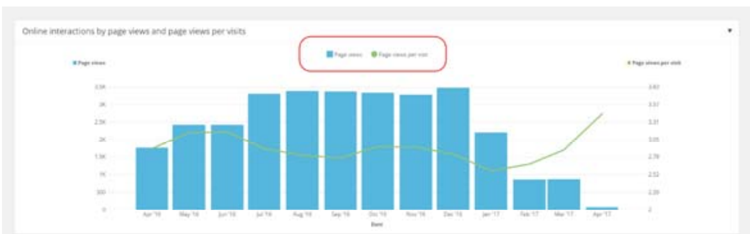
4. To clear the filter, click *Revert to default filter*.

Show or hide lines and bars in a line chart or graph

You can show or hide the lines in line charts and graphs displayed on the reports. If a report displays multiple metrics this can help you to focus more clearly on specific patterns and trends.

To show or hide lines in a line chart or graph:

1. In the Experience Analytics dashboard, open the report that you want to filter.
2. To show or hide data in a chart, click the metric that appears in the report. For example, in the Behavior tab, in the *Pages* report, you can show or hide two metrics:
   - Page views
   - Page views per visit



You can filter all Experience Analytics reports in this way.

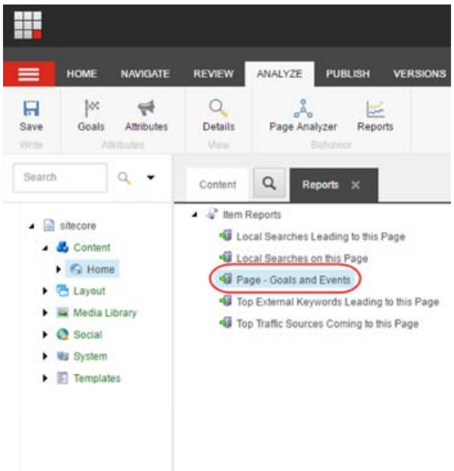
Send feedback about the documentation to docsite@sitecore.net.

Save and print an analytics report

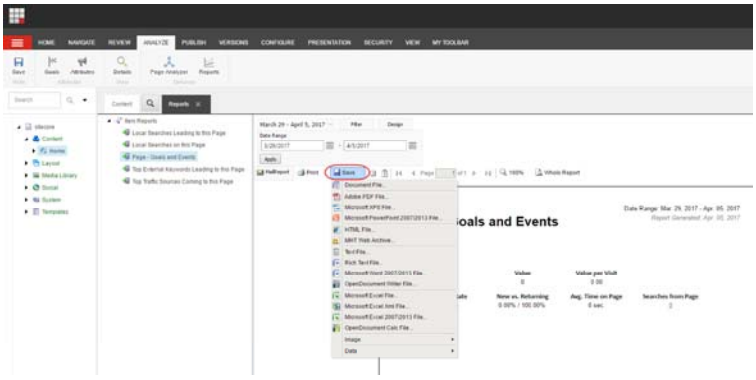
In Engagement Analytics, you can save a report in a number of different formats, such as Microsoft Word, PDF, XML, Excel, Text file format, and so on. You can also print out any analytics report.

To save and print a report:

1. In the Content Editor, on the ribbon, on the Analyze tab, in the Behavior group, click Reports.
2. In the right pane, expand the Item Reports tree, and click the report that you want to save or print, for example, the *Page - Goals and Events* report.



3. In the pane to the right-hand side of the Item Reports tree, to see the different formats that you can save the report in, click Save.



4. Select the file format that you want to save your report in. Specify the export settings for the relevant file format, and then click OK.
5. To print the report, click Print.

Send feedback about the documentation to docsite@sitecore.net.

The Acquisition tab reports

In the Experience Analytics dashboard, the Acquisition tab contains reports about what drives traffic to your website.

The Acquisition tab consists of six subtabs:

- Overview
- Channels
- Campaigns
- Keywords
- Paid overview
- Referring sites

Note

The tabs can contain duplicate reports showing the same data presented in a different way. For example, in a graph and in a pie chart. This topic does not describe each duplicate report.

You can find detailed technical information for every report, including duplicate reports and drill-down information, in the Sitecore Experience Analytics Standard Reports document.

The Experience Analytics glossary of terms can also help you to understand the reports.

Tabs	Reports
Overview Shows an overview of the key analytics reports on the Acquisition tab.	Top channels by visits The channels with the highest number of visits to your website. Top campaigns by visits The campaigns with the highest number of visits to your website.

Top campaign groups by visits

The campaign groups with the highest number of visits to your website.

Top referring sites by visits

The referring sites with the highest number of visits to your website.

Channels

Describes the traffic sources or channels, both online and offline, where the interactions on your website occurred.

Online and offline channels by visits

The online and offline channels with the highest number of visits to your website.

Top channels by visits

The online and offline channels with the highest number of visits to your website.

Top channels by value

The online and offline channels that have accumulated the highest number of engagement value points during visits.

Top channels by value per visit

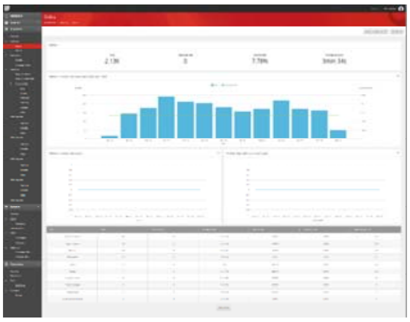
The online and offline channels that have provided the highest average engagement value generated per visit.

Table

Lists all channels, online and offline, and includes detailed metrics.

Online

Describes the online channels that drive traffic to your website.



Online

A summary of key metrics on the Online subtab showing visits, value per visit, bounce rate, and average duration for all online channels.

Online channels by visits and value per visit

The number of visits that your website has received within a specified period across all online channels, as well as the highest average engagement value generated per visit.

Online channels by value

The online channels that have accumulated the highest number of engagement value points during visits.

Online channels by conversions

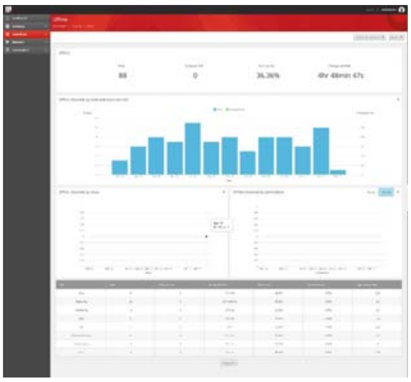
The online channels that have generated the most efficient high value visits on your website.

Table

Lists all online channel groups, and includes detailed metrics.

Offline

Describes the offline channels that drive traffic to your website



Offline

A summary of key metrics on the Offline subtab showing visits, value per visit, conversion rate percentage, bounce rate, and average duration.

Offline channels by visits and value per visit

The number of visits your website has received within a specified period, across all offline channels, as well as the highest average engagement value generated per visit.

Offline channels by value

The offline channels that have accumulated the highest number of engagement value points, generated by interactions from offline channels.

Offline channels by conversions

The offline channels that have generated the most efficient high value visits on your website.

Table

Lists all offline channel groups, and includes detailed metrics.

Campaigns

Top campaign groups by visits

Describes your marketing campaigns and shows how they are performing.

The campaign groups with the highest number of visits to your website.

Top campaign by visits

The campaigns with the highest number of visits to your website.

Top campaigns by visits

The number of interactions where given campaigns were registered, sorted by the total number of visits.

Top campaigns by value per visit

The campaigns that have provided the highest average engagement value generated per visit.

Top campaigns by bounces

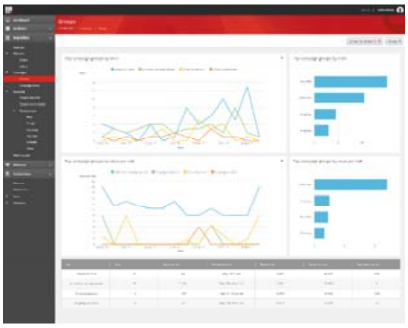
The campaigns with the highest number of bounces.

Table

Lists all campaigns that you are running, and includes detailed metrics.

Groups

Describes your grouped campaigns.



Top campaign groups by visits

The campaign groups with the highest number of visits to your website.

Top campaign groups by value per visit

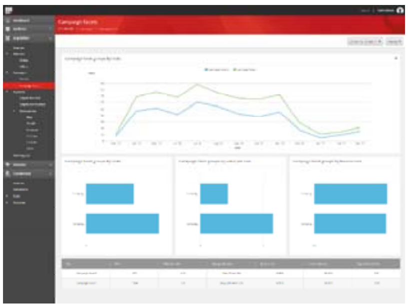
The campaign groups that have provided the highest average engagement value generated per visit.

Table

Lists all the campaign groups that you have, and includes detailed metrics.

Campaign facets

Describes your campaign facets, which you can use to classify your campaigns into meaningful groups.



Campaign facet groups by visits

The campaign facet groups with the highest number of visits to your website.

Campaign facet groups by value per visit

The campaign facet groups that have provided the highest average engagement value generated per visit.

Campaign facet groups by bounce rate

The campaign facet groups with the highest number of bounces.

Table

Lists all campaign facet groups that you have, and includes detailed metrics.

Keywords

Describes the keywords (words or phrases) that your contacts use to access or find your website.

Paid and organic search keywords by visits

The paid search keywords and organic search keywords with the highest number of visits to your website.

Top non-branded keywords

The number of visits to your website that originated from keywords that do not contain your website's brand name or some variation of it, as well as the highest average engagement value generated per visit.

Top branded keywords

The number of visits to your website that originated from keywords that included your website's brand name, as well as the average engagement value generated per visit.

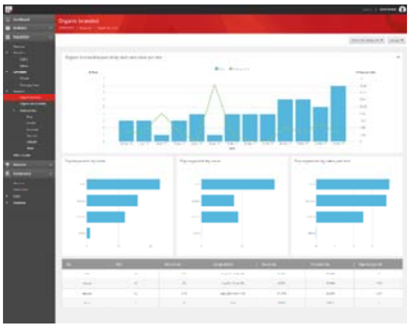
Top paid keywords

The number of visits to your website that originated from paid keywords, as well as the average engagement value generated per visit.

Organic branded

Organic branded keywords by visits and value per visit

Describes the keywords or search queries that are organic, and include your brand name or a variation of it.



The number of visits to your website that originated from searches that include keywords from your organic branded keywords, as well as the highest average engagement value generated per visit.

Top keywords by visits

The organic branded keywords with the highest number of visits to your website.

Top keywords by value

The organic branded keywords that have accumulated the highest number of engagement value points during visits.

Top keywords by value per visit

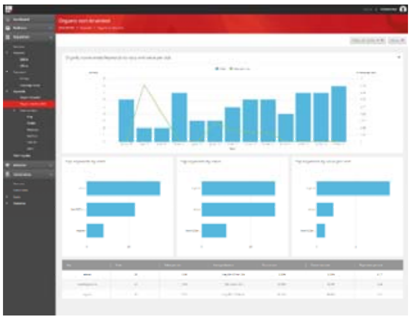
The organic branded keywords that have provided the highest average engagement value generated per visit.

Table

Lists all of the organic branded keywords, and includes detailed metrics.

Organic non-branded

Describes keywords or search queries that are organic, and do not include your brand name or any part of it.



Organic non-branded keywords by visits and value per visit

The number of visits to your website that originated from organic searches that did not include your website's brand name as a keyword, as well as the highest average engagement value generated per visit.

Top keywords by visits

The organic, non-branded keywords with the highest number of visits to your website.

Top keywords by value

The organic non-branded keywords that have accumulated the highest number of engagement value points during visits.

Top keywords by value per visit

The organic, non-branded keywords that have provided the highest average engagement value generated per visit.

Table

Lists all of the organic, non-branded keywords, and includes detailed metrics.

Paid overview

Describes your paid, search engine traffic/paid channels that were used by your contacts to find your website.

Paid keyword channels by visits

The paid keyword channels with the highest number of visits to your website.

Paid keywords by visits

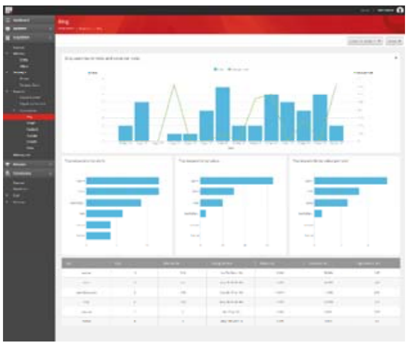
The paid keywords with the highest number of visits to your website.

Table

Lists all of the paid keywords, and includes detailed metrics.

Bing

Describes the search engine Bing as a paid channel.



Bing searches by visits and value per visits

The number of visits to your website that originated in Bing paid search, as well as the average engagement value generated per visit.

Top keywords by visits

The keywords that originated in Bing paid search with the highest number of visits to your website.

Top keywords by value

The keywords that originated in Bing paid search that have generated the most engagement value during visits.

Top keywords by value per visit

The keywords that originated in Bing paid search and that have provided the highest average engagement value generated per visit.

Table

Lists all paid keywords that originated in Bing, and includes detailed metrics.

Google

Describes the search engine Google as a paid channel.

Google searches by visits and value per visits

The number of visits to your website that originated in Google paid search, as well as the average engagement value generated per visit.

Top keywords by visits

The keywords that originated in Google paid search with the highest number of visits to your website.

Top keywords by value

The keywords that originated in Google paid search that have accumulated the highest number of engagement value points during visits.

Top keywords by value per visit

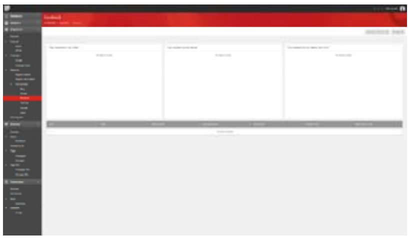
The keywords that originated in Google paid search and that have provided the highest average engagement value generated per visit.

Table

Lists all of the paid keywords that originated in Google, and includes detailed metrics.

Facebook

Describes the social networking site Facebook as a paid channel.



Facebook searches by visits and value per visits

The number of visits to your website that originated in Facebook paid search, as well as the average engagement value generated per visit.

Top keywords by visits

The keywords that originated in Facebook paid search with the highest number of visits to your website.

Top keywords by value

The keywords that originated in Facebook paid search that have accumulated the highest number of engagement value points during visits.

Top keywords by value per visit

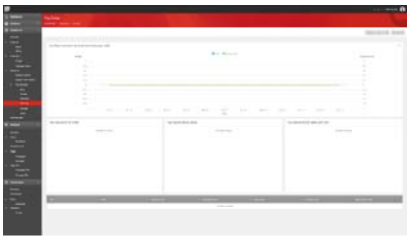
The keywords that originated in Facebook paid search and that have provided the highest average engagement value generated per visit.

Table

Lists all of the paid keywords that originated in Facebook, and includes detailed metrics.

YouTube

Describes the video-sharing website YouTube as a paid channel.



YouTube searches by visits and value per visits

The number of visits to your website that originated in YouTube paid search, as well as the average engagement value generated per visit.

Top keywords by visits

The keywords that originated in YouTube paid search with the highest number of visits to your website.

Top keywords by value

The keywords that originated in YouTube paid search and that have accumulated the highest number of engagement value points during visits.

Top keywords by value per visit

The keywords that originated from YouTube paid search and that have provided the highest average engagement value generated per visit.

Table

Lists all of the paid keywords that originated in YouTube, and includes detailed metrics.

LinkedIn

Describes the social networking site LinkedIn as a paid channel.

LinkedIn searches by visits and value per visits

The number of visits to your website that originated in LinkedIn paid search, as well as the average engagement value generated per visit.

Top keywords by visits

The keywords that originated in LinkedIn paid search with the highest number of visits to your website.

Top keywords by value

The keywords that originated in LinkedIn paid search and that have accumulated the highest number of engagement value points during visits.

Top keywords by value per visit

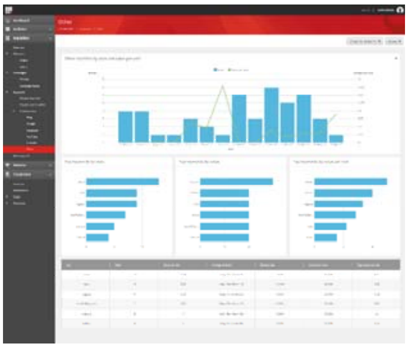
The keywords that originated in LinkedIn paid search and that have provided the highest average engagement value generated per visit.

Table

Lists all of the paid keywords that originated in LinkedIn, and includes detailed metrics.

Other

Describes all other paid channel providers.



Other searches by visits and value per visit

The total number of visits to your website that originated in ‘other’ providers’ paid search, as well as the average engagement value generated per visit.

Top keywords by visits

The keywords that originated in any of the ‘other’ providers' paid search, with the highest number of visits to your website.

Top keywords by value

The keywords that originated in ‘other’ providers' paid search and that have accumulated the highest number of engagement value points during visits.

Top keywords by value per visit

The keywords that originated in ‘other’ providers' paid search and that have the highest engagement value generated per visit.

Table

Lists all of the paid keywords that originated in ‘other’ providers’ searches, and includes detailed metrics.

Referring sites

Describes the external websites that refer contacts to your website.

Top referring sites by visits

The referring sites with the highest number of visits to your website.

Top referring sites by value

The referring sites that have accumulated the highest number of engagement value points during visits.

Top referring sites by value per visit

The referring sites that have provided the highest average engagement value generated per visit.

Table

Lists all of the referring sites, and includes detailed metrics.

Send feedback about the documentation to docsite@sitecore.net.

The Audience tab reports

In the Experience Analytics dashboard, the Audience tab contains reports that help you gain a better understanding of who your contacts are, by analyzing how they engage with your website.

The Audience tab consists of five subtabs:

- Overview
- Devices
- Languages
- Locations
- Pattern matches

Note

The tabs can contain duplicate reports showing the same data presented in a different way. For example, in a graph and in a pie chart. This topic does not describe each duplicate report.

You can find detailed technical information for every report, including duplicate reports and drill-down information, in the Sitecore Experience Analytics Standard Reports document.

The Experience Analytics glossary of terms can also help you to understand the reports.

Tabs	Reports
Overview Shows an overview of the key analytics reports on the Audience tab. 	Online interactions by visits and value per visit The number of visits your website has received within a specified period, as well as the average engagement value generated per visit. Top countries by visits The countries with the highest number of visits to your website. Top cities by visits The cities with the highest number of visits to your website. Pattern matches by visits The number of visits where contacts have matched with a given pattern card. Top languages by visits The languages in your website that your contacts have browsed in, and that have the highest number of visits.
Devices Describes the different device types and device models that access your website. 	Top device types by visits The device types with the highest number of visits to your website. Top device types by value per visit The device types that have provided the highest engagement value generated per visit. Top device types by bounces The device types with the highest number of bounces. Top device models by value per visit The device models that have provided the highest average engagement value generated per visit. Table Lists all the device types that visit your website and includes detailed metrics.
Device models Describes specific device models that interact with your website. 	Top device models by visits The device models with the highest number of visits to your website. Device models by value per visit The device models that have provided the highest average engagement value generated per visit. Device models by bounces The device models with the highest number of bounces. Device models by page views per visit The average total number of page views per visit of all device models. Table Lists all the device models that visit your website and includes detailed metrics
Languages Describes the languages that are most common on your website. 	Top languages by visits The languages on your website that your contacts have browsed in, and that have the highest number of visits. Top languages by value The languages on your website that your contacts have browsed in, and that have accumulated the highest number of engagement value points during visits. Top languages by value per visit The languages on your website that your contacts have browsed in, and that have provided the highest average engagement value generated per visit. Table Lists all the languages that your website is in, and includes detailed metrics.
	Top countries by visits

Locations

Describes specific geographical data of your website contacts.

The countries with the highest number of visits to your website.

Top regions by visits

The regions with the highest number of visits to your website.

Top cities by visits

The cities with the highest number of visits to your website.

Countries

Describes specific countries your contacts access your website from.

Top countries by visits

The countries with the highest number of visits to your website.

Top countries by value

The countries that have accumulated the highest number of engagement value points during visits.

Top countries by value per visit

The countries that have provided the highest average engagement value generated per visit.

Table

Lists all countries that visit your website and includes detailed metrics.

Regions

Describes specific regions your contacts access your website from.

Top regions by visits

The regions with the highest number of visits to your website.

Top regions by value

The regions that have accumulated the highest number of engagement value points during interactions.

Top regions by value per visit

The regions that have provided the highest average engagement value generated per visit.

Table

Lists all regions that visit your website and includes detailed metrics.

Cities

Describes specific cities your contacts access your website from.

Top cities by visits

The cities with the highest number of visits to your website.

Top cities by value

The cities that have accumulated the highest number of engagement value points during visits.

Top cities by value per visit

The cities that have provided the highest average engagement value generated per visit.

Table

Lists all cities that visit your website and includes detailed metrics.

Pattern matches

Describes how often your contacts match up with personas or profile cards.

Top pattern matches by visits

The pattern cards with the highest number of visits.

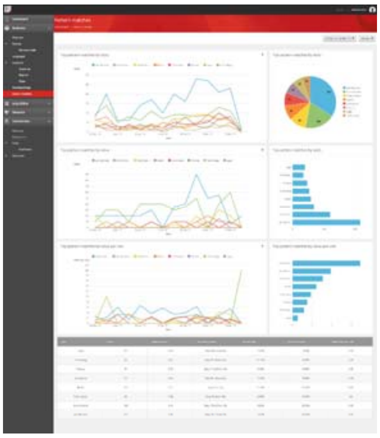
Top pattern matches by value

The pattern cards that contacts have matched with, and that have accumulated the highest number of engagement value points.

Top pattern matches by value per visit

The pattern cards that contacts have matched with, and that have provided the highest average engagement value generated per visit.

Table

	Lists all pattern cards that contacts have matched with and includes detailed metrics.

Send feedback about the documentation to docsite@sitecore.net.

The Behavior tab reports

In Experience Analytics, the Behavior tab contains reports that help you analyze the behavior of your contacts.

The Behavior tab consists of five subtabs:

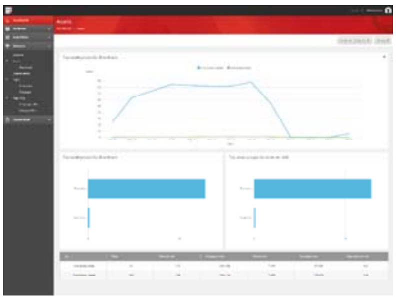
- Overview
- Assets
- Internal Search
- Pages
- Page URLs

Note

The tabs can contain duplicate reports showing the same data presented in a different way. For example, in a graph and in a pie chart. This topic does not describe each duplicate report.

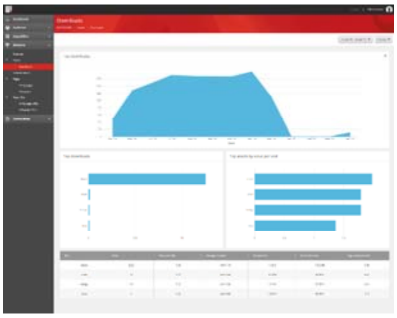
You can find detailed technical information for every report, including duplicate reports and drill-down information, in the Sitecore Experience Analytics Standard Reports document.

The Experience Analytics glossary of terms can also help you to understand the reports.

Tabs	Reports
Overview Shows an overview of the key analytics reports on the Behavior tab. 	Top downloaded assets The assets that your contacts have downloaded the most. Top internal keywords The internal keywords used on the internal search on your website. Top entry pages The entry pages that a contact used to visit your website. Top outcomes by visits The outcomes that have resulted from your contacts visiting your websites.
Assets Describes your marketing assets or content used to attract contacts to your website and increase their engagement with your organization. 	Top asset groups by downloads The most downloaded assets within a specific asset group. Top asset groups by value per visit The asset groups that have provided the highest engagement value generated per visit. Table Lists all the asset groups on your website, and includes detailed metrics.

Downloads

Describes your specific assets, their download activity, and their value.



Top downloads

The assets that your contacts have downloaded the most from your website.

Top assets by value per visit

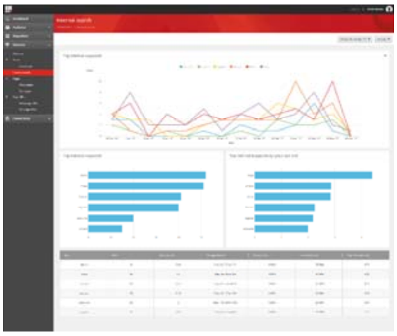
The specific assets that have provided the highest engagement value generated per visit.

Table

Lists the specific assets that your contacts have downloaded from your website, and includes detailed metrics.

Internal search

Describes how your contacts use the internal or local search on your website.



Top internal keywords

The internal keywords used on the internal search on your website.

Top internal keywords by value per visit

The internal keywords used on the internal search on your website that have provided the highest engagement value generated per visit.

Table

Lists the keywords used on the internal search on your website, and includes detailed metrics.

Pages

Describes the entry and exit pages, as well as page views on your website.

Online interactions by page views and page view per visits

The number of page views for a given page, and the average number of website pages a visitor views per visit.

Top entry pages by visits

The entry pages used by your contacts with the highest number of visits.

Top exit pages by visits

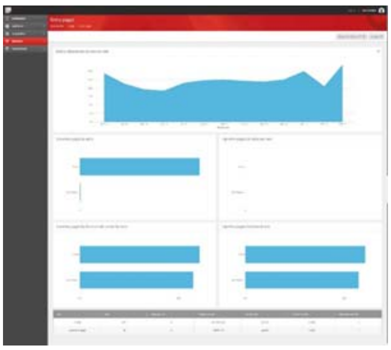
The exit pages used by your contacts with the highest number of visits.

Table

Lists all entry and exit pages used to enter or leave your website and includes detailed metrics.

Entry pages

Describes which pages your contacts open as their first page when visiting your website.



Online interactions by bounce rate

The percentage of contacts who only view one page and then leave your website.

Top entry pages by visits

The entry pages used by your contacts with the highest number of visits.

Top entry pages by value per visit

The entry pages used by your contacts that have provided the highest average engagement value generated per visit.

Top entry pages by bounce rate sorted by visits

The entry pages used by your contacts with the highest number of visits and the associated bounce rates.

Top entry pages by bounce rate

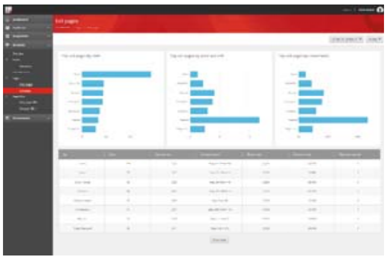
The entry pages used by your contacts who only view one page and then leave your website.

Table

Lists all the entry pages used to enter your website and includes detailed metrics.

Exit pages

Describes which pages your contacts visit when they leave your website.



Top exit pages by visits

The exit pages used by your contacts, with the highest number of visits.

Top exit pages by value per visit

The exit pages used by your contacts that have provided the highest average engagement value generated per visit.

Top exit pages by conversions

The exit pages used by your contacts that have generated the most efficient high value visits on your website.

Table

Lists all the exit pages used to enter your website and includes detailed metrics.

Page URLs

Describes the entry and exit page URLs.

Online interactions by page views and page views per visits

The number of page views for a given URL and the average number of URLs a visitor views per visit.

Top entry page URLs by visits

The entry page URLs used by your contacts with the highest number of visits to your website.

Top exit page URLs by visits

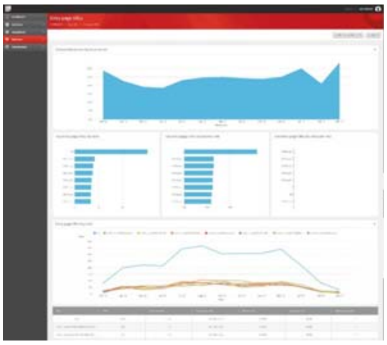
The exit page URLs used by your contacts with the highest number of visits to your website.

Table

Lists all the entry and exit page URLs used by your contacts and includes detailed metrics.

Entry page URLs

Describes the entry page URLs, or the URLs of the first page of an interaction with your contact.



Online interactions by bounce rate

The percentage of website visitors who only view one page and then leave your website.

Top entry page URLs by visits

The entry page URLs with the highest number of visits to your website.

Top entry page URLs by bounce rate

The entry page URLs with the highest number of bounces.

Top entry page URLs by value per visit

The entry page URLs used by your contacts to enter your website that have provided the highest average engagement value generated per visit.

Entry page URLs by visits

The entry page URLs with the highest number of visits to your website.

Table

Lists all entry page URLs used to enter your website and includes detailed metrics.

Exit page URLs

Describes the exit page URLs, or the last accessed URL when contacts leave your website.

Top exit page URLs by visits

The exit page URLs with the highest number of visits to your website.

Top exit page URLs by value per visit

The exit page URLs used by your contacts when they leave your website that have provided the highest average engagement value generated per visit.

Top exit page URLs by conversions

The exit page URLs that have generated the most efficient high value visits on your website.

Exit page URLs by visits

The exit page URLs with the highest number of visits to your website.

Table

Lists all exit page URLs that your contacts used to leave your website, and includes detailed metrics.

Send feedback about the documentation to docsite@sitecore.net.

The Conversions tab reports

In the Experience Analytics dashboard, the Conversions tab contains reports about contacts that complete target actions and achieve particular goals. These reports help you to assess which target actions generate the most value and to understand how well your marketing efforts are working.

The Conversions tab consists of four subtabs:

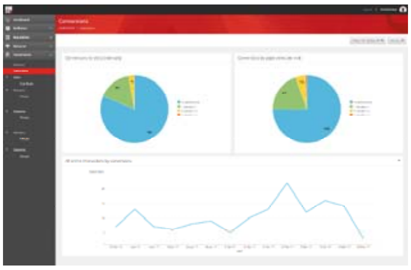
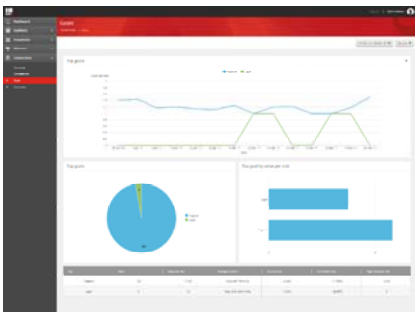
- Overview
- Conversions
- Goals
- Outcomes

Note

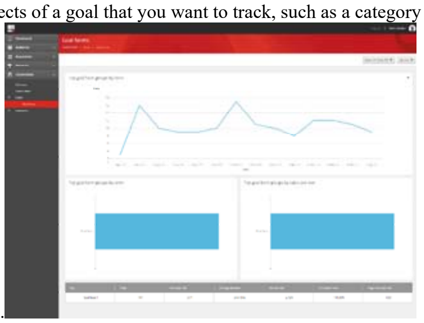
The tabs can contain duplicate reports showing the same data presented in a different way. For example, in a graph and in a pie chart. This topic does not describe each duplicate report.

You can find detailed technical information for every report, including duplicate reports and drill-down information, in the Sitecore Experience Analytics Standard Reports document.

The Experience Analytics glossary of terms can also help you to understand the reports.

Tabs	Reports
Overview Shows an overview of the key analytics reports on the Conversions tab. 	Online interactions by conversions and conversion rates The percentage of interactions that have a conversion as well as the total number of conversions. Top goals The top goals converted on your site.
Conversions Shows an overview of how often your contacts are converting on your sites. 	Conversions by visits (intervals) The number of times your contacts are converting per visit. Conversions by page views per visit The number of times your contacts are converting, by page views. All online interactions by conversions The number of interactions where conversions took place one or more times.
Goals Describes the goals that help you measure the effectiveness of your campaigns. 	Top goals The goals achieved by your contacts. Top goal by value per visit The goals that have provided the highest average engagement value generated per visit. Table Lists all goals and includes detailed metrics.
Goal facets	Top goal facet groups by visits

Describes the different aspects of a goal that you want to track, such as a category or description of related goals.



The goal facet groups with the highest number of visits to your website.

Top goal facet groups by value per visit

The goal facet groups that have provided the highest average engagement value generated per visit.

Table

Lists all goal facets and includes detailed metrics.

Outcomes

Describes [outcomes] or business-significant results of one or more interactions between a contact and your organization.

Top outcome groups by visits

The outcome groups with the highest number of visits to your website.

Top outcomes by visits

The outcomes with the highest number of visits to your website.

Outcomes by visits

The outcomes with the highest number of visits to your website.

Outcomes by value

The outcomes that have accumulated the highest number of engagement value points during visits.

Top outcomes by value per visit

The outcomes that have provided the highest average engagement value generated per visit.

Table

Lists all specific outcomes and includes detailed metrics.

Groups

Describes outcomes, further divided into groups.

Top outcomes groups by visit

The outcome groups with the highest number of visits to your website.

Top outcomes groups by value per visit

The outcome groups that have provided the highest average engagement value generated per visit.

Table

Lists all outcome groups and includes detailed metrics.

Send feedback about the documentation to docsite@sitecore.net.

The Dashboard tab reports

In the Experience Analytics dashboard, to help you identify general marketing trends on your website, the Dashboard tab contains reports that gather the most important performance indicators from all of the other tabs.

Note

The tabs can contain duplicate reports showing the same data presented in a different way. For example, in a graph and in a pie chart. This topic does not describe each duplicate report.

You can find detailed technical information for every report, including duplicate reports and drill-down information, in the Sitecore Experience Analytics Standard Reports document.

The Experience Analytics glossary of terms can also help you to understand the reports.

Tab	Reports
Dashboard Shows an overview of the key analytics reports on the Dashboard tab.	Online interactions A summary of key metrics of your entire website, showing visits, value per visit, conversion rate, and bounce rate. Online interactions by visits and value per visit

The number of visits your website has received within a specified period, as well as the average engagement value generated per visit.

Channel groups by visits

The channel groups with the highest number of visits to your website.

Top campaigns by value

The campaigns that have accumulated the highest number of engagement value points during visits.

Top pattern matches by value per visit

The pattern cards that contacts have matched with, and that have provided the highest average engagement value generated per visit.

Top goals by conversions

The goals that have generated the most efficient high-value visits on your website.

Send feedback about the documentation to docsite@sitecore.net.

The dimensions and metrics in Experience Analytics reports

Sitecore developers and administrators use dimensions and metrics when creating Experience Analytics reports.

You can use dimensions to group the data used in Experience Analytics reports, for example, in an interaction, campaigns triggered, entry pages visited, or downloads made.

In Sitecore, there are seven standard metrics that measure the different attributes of each dimension:

- Visit
- Value
- PageViews
- TimeOnSite
- Bounces
- Conversions
- Count

For example, in the *Campaign* dimension, the *Visits* metric counts the number of visits (interactions) where a given campaign was registered. What the metrics measure can have a slightly different context depending on each dimension.

When you create Experience Analytics reports, you can use the following dimensions in combination with a standard set of metrics to analyze the activity on your website:

Dimension	Dimension	Dimension	Dimension
Downloads	Page views	Page	Asset facet
Language	Search keyword	Entry page URL	Asset group
Channel	Local search keyword	Entry pages	Goal facet
Channel group	City	Exit page URL	Goal facet group
Channel type	Region	Exit pages	Campaign facet
Goal	Country	Email	Campaign facet group
Conversions	Referring sites	Language specific email	Device type
Campaign	Pattern	Email manager	Device model
Campaign group	Page URL	Asset	

Downloads

Use the Downloads dimension to analyze the interactions where a file download occurred.

Metric	Description

Visits	Number of interactions where there was one or more downloads.
Value	Accumulated value from interactions where there was one or more downloads.
Conversions	Number of conversions from interactions where there was one or more downloads.
Time on site	Interaction duration where there was one or more downloads.
Page views	Number of page views where there was one or more downloads.
Bounces	Number of interactions where an item was downloaded one or more times and where only one page was viewed.
Count	Number of times the item was downloaded.

Language

The Language dimension refers to the language that a contact chooses to view the pages on a website in.

The Language dimension tracks the language of the initial interaction, not the language of the individual web pages. For example, you can see how much value is generated by English content versus Danish content.

Metric	Description
Visits	Number of interactions where a given language was the primary language.
Value	Accumulated value from interactions where a given language was chosen.
Conversions	Number of conversions from interactions where a given language was chosen.
Time on site	Interaction duration of interactions where a given language was chosen.
Page views	Page views during interactions where a given language was chosen.
Bounces	Number of interactions where a given language was chosen and where exactly one page was viewed.

Channel

Channels are the marketing channels or traffic sources where the interactions on your website occurred.

Use the Channel dimension to analyze which channels perform best on your website. For example, which channels generate the most engagement value.

Metric	Description
Visits	Number of interactions from a given channel.
Value	Engagement value generated by interactions from a given channel.
Conversions	Number of conversions from a given channel.
Time on site	Interaction duration of interactions from a given channel.
Page views	Number of page views in interactions from a given channel.
Bounces	Number of interactions from a given channel where only one page was viewed.

Channel group

Interactions from every channel in a channel group are categorized at the group level. You can drill down from channel groups to channels.

Metric	Description
Visits	Number of interactions where the given interaction came from a channel within a specific channel group.

Value	Accumulated value from interactions where the given interaction came from a channel within a specific channel group.
Conversions	Number of conversions from interactions where the given interaction originated from a channel within a specific channel group.
Time on site	Interaction duration where the interaction originated from a channel within a specific channel group.
Page views	Number of page views from interactions where the given interaction came from a channel within a specific channel group.
Bounces	Number of interactions where the given interaction came from a channel within a specific channel group and where only one page was viewed.

Channel type

By default, you can use two basic types of channels to interact with your contacts: online or offline. Interactions from every channel are summarized at the channel type level. You can drill down from channel type to channel group or channel. Use the Channel type dimension to analyze whether online or offline channels perform best. For example, which channel types generate the most engagement value.

Metric	Description of usage
Visits	Number of interactions where the given interaction came from a channel within a specific channel type.
Value	Accumulated value from interactions where the given interaction came from a channel within a specific channel type.
Conversions	Number of conversions from interactions where the given interaction came from a channel within a specific channel type.
Time on site	Interaction duration where the interaction originated from a channel within a specific channel type.
Page views	Number of page views from interactions where the given interaction came from a channel within a specific channel type.
Bounces	Number of interactions where the given interaction came from a channel within a specific channel type and where only one page was viewed.
Count	Number of interactions where the given interaction came from a channel within a specific channel type.

Goal

A goal is a specific activity (page event) on a website that indicates success to the site owner. For example, filling in and submitting a form could be a goal. Use the Goal dimension to analyze the goals achieved during interactions.

Metric	Description
Visits	Number of times a given goal has been triggered one or more times in an interaction.
Value	Accumulated value from interactions where a given goal has been triggered one or more times in an interaction.
Conversions	Number of conversions from interactions where a given goal has been triggered one or more times in an interaction.
Time on site	Interaction duration where a given goal has been triggered one or more times.
Page views	Page views from interactions where a given goal has been triggered one or more times in an interaction.
Bounces	Number of times a given goal has been triggered one or more times in an interaction and where only one page was viewed.
Count	Number of times a goal has been triggered.

Conversions

Conversions occur when a contact completes a target action on a website and triggers a goal that is associated with it. For example, submitting a web form to request a product demo could be a conversion.

Metric	Description

Visits	Number of interactions where a conversion took place one or more times (grouped by interval).
Value	Accumulated value from interactions where a conversion took place one or more times (grouped by interval).
Conversions	Number of conversions from interactions where a conversion took place one or more times (grouped by interval).
Time on site	Interaction duration from interactions where a conversion took place one or more times (grouped by interval).
Page views	Page views from interactions where a conversion took place one or more times (grouped by interval).
Bounces	Number of interactions where a conversion took place one or more times (grouped by interval) and where only one page was viewed.

Campaign

Campaigns are advertising initiatives designed to encourage people to visit a website and can appear in different media across a specific time frame. A contact can trigger several campaigns during an interaction, but the campaign dimension measures the first campaign that was triggered.

You use the Campaign dimension to analyze and assess the campaigns on your website.

Metric	Description
Visits	Number of interactions where a given campaign was registered.
Value	Accumulated value from interactions where a given campaign was registered.
Conversions	Number of conversions from interactions where a campaign was registered.
Time on site	Interaction duration from interactions where a campaign was registered.
Page views	Number of page views from interactions where a campaign was registered.
Bounces	Number of interactions where a given campaign was registered and where only one page was viewed.

Campaign group

A campaign group is a way of classifying individual campaigns under a higher-level campaign group that contains multiple campaigns across different channels. The Campaign group dimension functions as a tag to identify the campaigns that belong to a particular campaign group. Interactions from every campaign are summarized at the campaign-group level. You can drill down from a campaign group to individual campaigns.

Metric	Description
Visits	Number of interactions where the given interaction came from a campaign within a specific campaign group.
Value	Accumulated value from interactions where the given interaction came from a campaign within a specific campaign group.
Conversions	Number of conversions from interactions where the given interaction came from a campaign within a specific campaign group.
Time on site	Interaction duration of interactions where the given interaction originated from a campaign within a specific campaign group.
Page views	Number of page views from interactions where the given interaction came from a campaign within a specific campaign group.
Bounces	Number of interactions where the given interaction came from a campaign within a specific campaign group and where only one page was viewed.

Page views

The Page views dimension enables you to compare interactions with the number of page views. For example, this can help you to understand if there is any value connected to the number of pages that were viewed during an interaction.

Metric	Description
Visits	Number of interactions where several page views took place (grouped by interval).
Value	Accumulated value from interactions where several page views took place (grouped by interval).
Conversions	Number of conversions from interactions where several page views took place (grouped by interval).
Time on site	Interaction duration of interactions where several page views took place (grouped by interval).
Page views	Number of page views from interactions where several page views took place (grouped by interval).
Bounces	Number of interactions where several page views took place and where only one page was viewed (grouped by interval).

Search keyword

The search keyword is the word or phrase that a contact initially enters into a search engine when looking for a website. The search is the starting point of the interaction. This dimension is not case sensitive, so the interactions originating from searches such as "sitecore" and "Sitecore" both appear in reports as "sitecore". Use the Search keywords dimension to analyze the actions and behavior of contacts who found your site using different search keywords.

Note

The search keyword dimension relies on referral data. This information is not always available, and depends on the browser or search engine that has been configured.

Metric	Description
Visits	Number of interactions where a given search keyword was registered.
Value	Accumulated value from interactions where a given search keyword was registered.
Conversions	Number of conversions from interactions where a given search keyword was registered.
Time on site	Interaction duration where a given search keyword was registered.
Page views	Number of page views from interactions where a given search keyword was registered.
Bounces	Number of interactions where a given search keyword was registered and where only one page was viewed.
Count	Actual number of times the keyword was registered.

Local search keyword

Local search keywords are the words or phrases that a contact enters into the Search field on your website. This dimension is not case sensitive, so the interactions searching for "sitecore" and "Sitecore" appear in reports as "sitecore". The local search keyword is triggered by the Search page event. Use the Local search keyword dimension to analyze the actions and behavior of contacts that use different local search keywords on your website.

Metric	Description
Visits	Number of interactions where a given local search keyword was registered one or more times.
Value	Accumulated value from interactions where a given local search keyword was registered one or more times.
Conversions	Number of conversions from interactions where a given local search keyword was registered one or more times.
Time on site	Interaction duration where a given local search keyword was registered one or more times.
Page views	Number of page views from interactions where a given local search keyword was registered one or more times.
Bounces	Number of interactions where a given local search keyword was registered one or more times and where only one page was viewed.
Count	Actual number of times the local search keyword was registered.

City

Use the City dimension to analyze the interactions coming from different cities using Geo IP data. For example, you can compare the behavior of contacts in interactions from different cities.

Metric	Description
Visits	Number of interactions where a given city was registered.
Value	Accumulated value from interactions where a given city was registered.
Conversions	Number of conversions from interactions where a given city was registered.
Time on site	Interaction duration of interactions where a given city was registered.
Page views	Number of page views from interactions where a given city was registered.
Bounces	Number of interactions where a given city was registered and where only one page was viewed.

Region

Use the Region dimension to analyze the interactions coming from different regions using Geo IP data. For example, you can compare the behavior of contacts in interactions from different regions. You can drill down from region to city.

Metric	Description
Visits	Number of interactions where a given region was registered.
Value	Accumulated value from interactions where a given region was registered.
Conversions	Number of conversions from interactions where a given region was registered.
Time on site	Interaction duration where a given region was registered.
Page views	Number of page views from interactions where a given region was registered.
Bounces	Number of interactions where a given region was registered and where only one page was viewed.

Country

Use the Country dimension to analyze the interactions of contacts coming from different countries using Geo IP data. For example, you can compare the behavior in interactions from different countries. You can drill down from country to region or city.

Metric	Description
Visits	Number of interactions where a given country was registered.
Value	Accumulated value from interactions where a given country was registered.
Conversions	Number of conversions from interactions where a given country was registered.
Time on site	Interaction duration where a given country was registered.
Page views	Number of page views from interactions where a given country was registered.
Bounces	Number of interactions where a given country was registered and where only one page was viewed.

Referring sites

Use the Referring sites dimension to analyze the behavior of contacts coming to your website from referring sites.

Metric	Description

Visits	Number of interactions where a given referring site was registered.
Value	Accumulated value from interactions where a given referring site was registered.
Conversions	Number of conversions from interactions where a given referring site was registered.
Time on site	Interaction duration where a given referring site was registered.
Page views	Number of page views from interactions where a given referring site was registered.
Bounces	Number of interactions where a given referring site was registered and where only one page was viewed.

Pattern

A pattern is a way to categorize an interaction using predictive personalization. Use the Pattern dimension to analyze the behavior of interactions matched to patterns across all profiles. This dimension enables you to see which patterns have the most matches, which patterns do not have matches, or which have conversions.

Metric	Description
Visits	Number of interactions where a given pattern card was matched.
Value	Accumulated value from interactions where a given pattern card was matched.
Conversions	Number of conversions from interactions where a given pattern card was matched.
Time on site	Interaction duration where a given pattern card was matched.
Page views	Number of page views from interactions where a given pattern card was matched.
Bounces	Number of interactions where a given pattern card was matched and where only one page was viewed.
Count	Actual number of times where a given pattern card was matched.

Page URL

Page URL refers to any URL on your website. Use the Page URL dimension to analyze and assess the performance of your website URLs. This dimension provides a breakdown of the raw URLs requested by contacts (path and query). The dimension is not case sensitive, which means that, for example, /Products and /products are displayed in your reports as one URL: /products.

Metric	Description
Visits	Number of interactions where a given page URL was registered one or more times.
Value	Accumulated value from interactions where a given page URL was registered one or more times.
Conversions	Number of conversions from interactions where a given page URL was registered one or more times.
Time on site	Interaction duration where a given page URL was registered one or more times.
Page views	Number of page views for the given page URL.
Bounces	Number of interactions where a given page URL was registered one or more times and where only one page was viewed.
Count	Page views for a given page URL in seconds.

Page

Page refers to any page on your Sitecore website. Use the Page dimension to analyze and assess the performance of your website pages.

Metric	Description

Visits	Number of interactions where a given page was registered one or more times.
Value	Accumulated value from interactions where a given page was registered one or more times.
Conversions	Number of conversions from interactions where a given page was registered one or more times.
Time on site	Interaction duration where a given page was registered one or more times.
Page views	Number of page views for the given page.
Bounces	Number of interactions where a given page was registered one or more times and where only one page was viewed.
Count	Page views for a given page in seconds.

Entry page URL

An entry page URL is the URL of the first page of an interaction. Use the Entry page URL dimension to analyze the activity of contacts that enter your website on a specific entry page URL.

Metric	Description
Visits	Number of interactions where a given entry page URL was registered one or more times.
Value	Accumulated value from interactions where a given entry page URL was registered one or more times.
Conversions	Number of conversions from interactions where a given entry page URL was registered one or more times.
Time on site	Interaction duration where a given entry page URL was registered one or more times.
Page views	Number of page views from the given entry page URL.
Bounces	Number of interactions where a given entry page URL was registered one or more times and where only one page was viewed.
Count	Page views for a given entry page URL in seconds.

Entry pages

An entry page is the first page a contact opens when visiting a website. Use the Entry pages dimension to analyze the activity of contacts that enter your website on specific entry pages.

Metric	Description
Visits	Number of interactions where a given entry page was registered one or more times.
Value	Accumulated value from interactions where a given entry page was registered one or more times.
Conversions	Number of conversions from interactions where a given entry page was registered one or more times.
Time on site	Interaction duration where a given entry page was registered one or more times.
Page views	Number of page views from the given entry page.
Bounces	Number of interactions where a given entry page was registered one or more times and where only one page was viewed.
Count	Page views for a given entry page in seconds.

Exit page URL

An exit page URL is the last URL accessed when a contact leaves a website, signifying the end of an interaction/session. Use the Exit page URL dimension to analyze the activity of contacts that leave your Sitecore website on a specific URL.

Metric	Description
Visits	Number of interactions where a given exit page was registered one or more times.
Value	Accumulated value from interactions where a given exit page was registered one or more times.
Conversions	Number of conversions from interactions where a given exit page was registered one or more times.
Time on site	Interaction duration where a given exit page was registered one or more times.
Page views	Number of page views from the given exit page.
Bounces	Number of interactions where a given exit page was registered one or more times and where only one page was viewed.
Count	Page views for a given exit page in seconds. Currently, it is not possible to measure duration for Exit page URLs, so this value is always zero.

Exit pages

An exit page is the last page accessed when a contact leaves a website, signifying the end of an interaction/session. Use the Exit pages dimension to analyze the activity of contacts or visitors that leave your Sitecore website on specific pages.

Metric	Description
Visits	Number of interactions where a given exit page was registered one or more times.
Value	Accumulated value from interactions where a given exit page was registered one or more times.
Conversions	Number of conversions from interactions where a given exit page was registered one or more times.
Time on site	Interaction duration where a given exit page was registered one or more times.
Page views	Number of page views from the given exit page.
Bounces	Number of interactions where a given exit page was registered one or more times and where only one page was viewed.
Count	Page views for a given exit page in seconds. Currently, it is not possible to measure duration for Exit pages, so this value is always zero.

Language specific email

A language-specific email is an email sent in a specific language. Use the Language-specific email dimension to analyze the interactions where contacts clicked a link in an email sent in a specific language.

Metric	Description
Visits	Number of interactions where a given language was used when sending an email message.
Value	Accumulated value from where a given language was used when sending an email message.
Conversions	Number of conversions from interactions where a given language was used when sending an email message.
Time on site	Interaction duration where a given language was used when sending an email message.
Page views	Page views from interactions where a given language was used when sending an email message.
Bounces	Number of interactions where a given language was used when sending an email message and where only one page was viewed.

Email

Use the Email dimension to analyze the interactions where contacts clicked a link in a specific email. You can drill down from email to a language specific email.

Metric	Description
Visits	Number of interactions from a given email.
Value	Accumulated value from interactions from a given email.
Conversions	Number of conversions from a given email.
Time on site	Interaction duration of interactions from a given email.
Page views	Page views from interactions from a given email.
Bounces	Number of interactions from a given email and where only one page was viewed.

Email manager

Email manager is the root manager used to send email messages. Use the Email manager dimension to analyze the interactions where contacts clicked a link in an email message from a specific root manager. You can drill down from email manager to email or to a language specific email.

Metric	Description
Visits	Number of interactions where a given email manager was used to send email messages.
Value	Accumulated value from interactions where a given email manager was used to send email messages.
Conversions	Number of conversions from interactions where a given email manager was used to send email messages.
Time on site	Interaction duration of interactions where a given email manager was used to send email messages.
Page views	Page views from interactions where a given email manager was used to send email messages.
Bounces	Number of interactions where a given email manager was used to send email messages and where only one page was viewed.

Asset

Marketing assets are the content that you use to attract contacts to your website and increase their engagement with your organization. If a marketing asset is a whitepaper, webinar, or other downloadable interactive content, you can use this dimension to track how frequently contacts consume your marketing assets.

Metric	Description
Visits	Number of interactions where the asset was downloaded one or more times within that visit.
Value	Accumulated value from interactions where the asset was downloaded one or more times.
Conversions	Number of conversions from interactions where the asset was downloaded one or more times.
Time on site	Interaction duration of interactions where the asset was downloaded one or more times.
Page views	Number of page views from interactions where the asset was downloaded one or more times.
Bounces	Number of interactions where the asset was downloaded one or more times within that visit and that visit only had one page view.
Count	Number of times the asset was downloaded.

Asset facet

An asset facet is a taxonomic grouping that describes characteristics of an asset that you want to track. Assets are typically media items, for example, logos. This dimension enables you to track how frequently contacts download assets of a certain defined type, or facet.

Metric	Description
Visits	Number of interactions where an asset belonging to the given asset facet was downloaded one or more times within that visit.
Value	Accumulated value from interactions where an asset belonging to the given asset facet was downloaded one or more times.
Conversions	Number of conversions from interactions where an asset belonging to the given asset facet was downloaded one or more times.
Time on site	Interaction duration of interactions where an asset belonging to the given asset facet was downloaded one or more times.
Page views	Number of page views from interactions where an asset belonging to the given asset facet was downloaded one or more times.
Bounces	Number of interactions where an asset belonging to the given asset facet was downloaded one or more times within that visit and that visit only had one page view.
Count	Number of times an asset belonging to the given asset facet was downloaded.

Asset group

An asset group is a taxonomic grouping that enables you to group marketing assets into related categories, for example, marketing collateral. This dimension provides you with information about how different asset groups are performing.

Metric	Description
Visits	Number of interactions where an asset belonging to the given asset facet group was downloaded one or more times within that visit.
Value	Accumulated value from interactions where an asset belonging to the given asset facet group was downloaded one or more times.
Conversions	Number of conversions from interactions where an asset belonging to the given asset facet group was downloaded one or more times.
Time on site	Interaction duration of interactions where an asset belonging to the given asset facet group was downloaded one or more times
Page views	Number of page views from interactions where an asset belonging to the given asset facet group was downloaded one or more times
Bounces	Number of interactions where an asset belonging to the given asset facet group was downloaded one or more times within that visit and that visit only had one page view.
Count	Number of times an asset belonging to the given asset facet group was downloaded

Goal facet

A goal facet describes different aspects of a goal that you want to track, for example, a category or description of related goals. This dimension allows you to track how often contacts convert on goals that have been tagged with a specific goal facet.

Metric	Description
Visits	Number of interactions where a goal that has a given goal facet assigned in classification was registered.
Value	Accumulated value from interactions where a goal that has a given goal facet assigned in classification was registered.
Conversions	Number of conversions from interactions where a goal that has a given goal facet assigned in classification was registered.
Time on site	Interaction duration from interactions where a goal that has a given goal facet assigned in classification was registered.

Page views	Number of page views from interactions where a goal that has a given goal facet assigned in classification was registered.
Bounces	Number of interactions where a goal that has a given goal facet assigned in classification was registered and that interaction only had one page view.
Count	Actual number of times a goal was triggered that has given goal facet assigned.

Goal facet group

A goal facet group is a taxonomy group that contains related goal facets. You can use this dimension to track information on how often these goals converted during interactions. From here, you can drill down to goal facets and any related goal facet child items.

Metric	Description
Visits	Number of interactions where a goal that has been assigned a goal facet which belongs to given goal facet group was registered.
Value	Accumulated value from interactions where a goal that has been assigned a goal facet which belongs to given goal facet group was registered.
Conversions	Number of conversions from interactions where a goal that has been assigned a goal facet which belongs to given goal facet group was registered.
Time on site	Interaction duration from interactions where a goal that has been assigned a goal facet having a given goal facet group was registered.
Page views	Number of page views from interactions where a goal that has a goal that has been assigned a goal facet having a given goal facet group was registered.
Bounces	Number of interactions where a goal that has been assigned a goal facet which belongs to given goal facet group was registered and that interaction has only one page view.
Count	Actual number of times a goal was triggered that has a goal facet assigned which belongs to given goal facet group.

Campaign facet

A campaign facet is a customizable taxonomy tag that allows you to classify your campaigns into meaningful groups, for example, by geographic region or business unit. This dimension enables you to track how contacts interact with different types of campaign activities.

Metric	Description
Visits	Number of interactions where a campaign belonging to a given campaign facet was registered.
Value	Accumulated value from interactions where a given campaign belonging to a given campaign facet was registered.
Conversions	Number of conversions from interactions where the campaign belonging to a given campaign facet was registered.
Time on site	Interaction duration from interactions where the campaign belonging to a given campaign facet was registered.
Page views	Number of page views from interactions where the campaign belonging to a given campaign facet was registered.
Bounces	Number of interactions where a given campaign belonging to a given campaign facet was registered and that interaction only had one page view.

Campaign facet group

A campaign facet group organizes related campaign facets. You can use this dimension to track how contacts interact with campaign activities that have been tagged as belonging to this campaign facet group. From this group, you can drill down to campaign facets and their subgroups.

Metric	Description
Visits	Number of interactions where a given campaign was registered at the interaction level, and that campaign has attached a campaign facet to, which belongs to a specific campaign facet group.

Value	Accumulated value from interactions where a given campaign was registered at the interaction level, and that campaign has a campaign facet attached to it, which belongs to a specific campaign facet group.
Conversions	Number of conversions from interactions where a given campaign was registered at the interaction level, and that campaign has a campaign facet attached to it, which belongs to a specific campaign facet group.
Time on site	Interaction duration from interactions where a given campaign was registered at the interaction level, and that campaign has a campaign facet attached to it, which belongs to a specific campaign facet group.
Page views	Number of page views from an interaction where a given campaign was registered at the interaction level and that campaign has a campaign facet attached to it, which belongs to a specific campaign facet group.
Bounces	Number of interactions where a given campaign was registered at the interaction level, and that campaign has a campaign facet attached to it, which belongs to a specific campaign facet group, and that interaction only had one page view.

Device type

The device type dimension measures how contacts interact with content from various devices, such as PCs, mobile phones and tablets. From here, you can drill down to device model.

Metric	Description
Visits	Number of interactions where a given device type was used to access content.
Value	Accumulated engagement value from interactions where a given device type was used to access content.
Conversions	Number of conversions from interactions where a given device type was used to access content.
Time on site	Interaction duration of interactions from a given device type.
Page views	Page views from interactions from a given device type.
Bounces	Number of interactions from a given device type where only one page was viewed.

Device model

The Device model dimension enables you to you track how contacts interact with content from specific device models, for example, Samsung Galaxy s5 or iPad 2.

Metric	Description
Visits	Number of interactions where a specific device model was used to access content.
Value	Accumulated engagement value from interactions where a specific device model was used to access content.
Conversions	Number of conversions from interactions where a specific device model was used to access content.
Time on site	Interaction duration of interactions from a specific device model.
Page views	Page views from interactions from a specific device model.
Bounces	Number of interactions from a specific device model where only one page was viewed.

The Experience Analytics reports for marketers

Sitecore Experience Analytics provides dashboards and reports for marketers and marketing analysts to identify patterns and trends in experience data. This data is collected from websites and other external data sources.

xAnalytics contains a number of reports that analyze the ways that contacts engage with your website. You can also create and configure custom reports in xAnalytics. In your daily activities as a marketer, some reports may be more relevant than others.

This information can help you analyze the amount of traffic on your website, as well as the quality of that traffic. You can use this information to refine and optimize your campaigns.

You can find the most popular reports for marketers in:

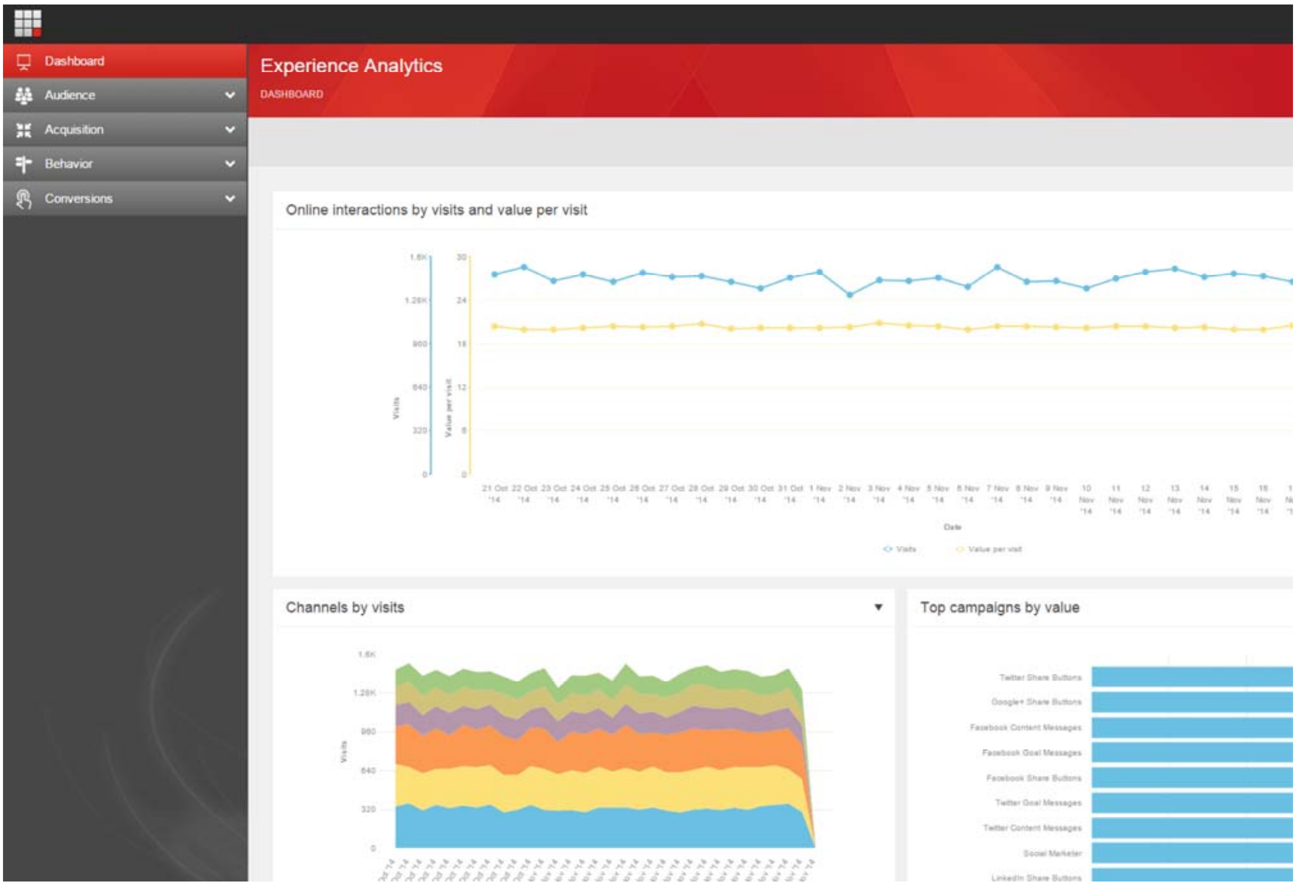
- The Dashboard
- The Audience tab
- The Acquisitions tab
- The Conversions tab

The Dashboard

The Experience Analytics Dashboard contains five charts that provide general overviews of contact activities on your website. You can use the Dashboard to identify general marketing trends on your website. You can find these charts under other categories in xAnalytics.

The charts included in the Dashboard are:

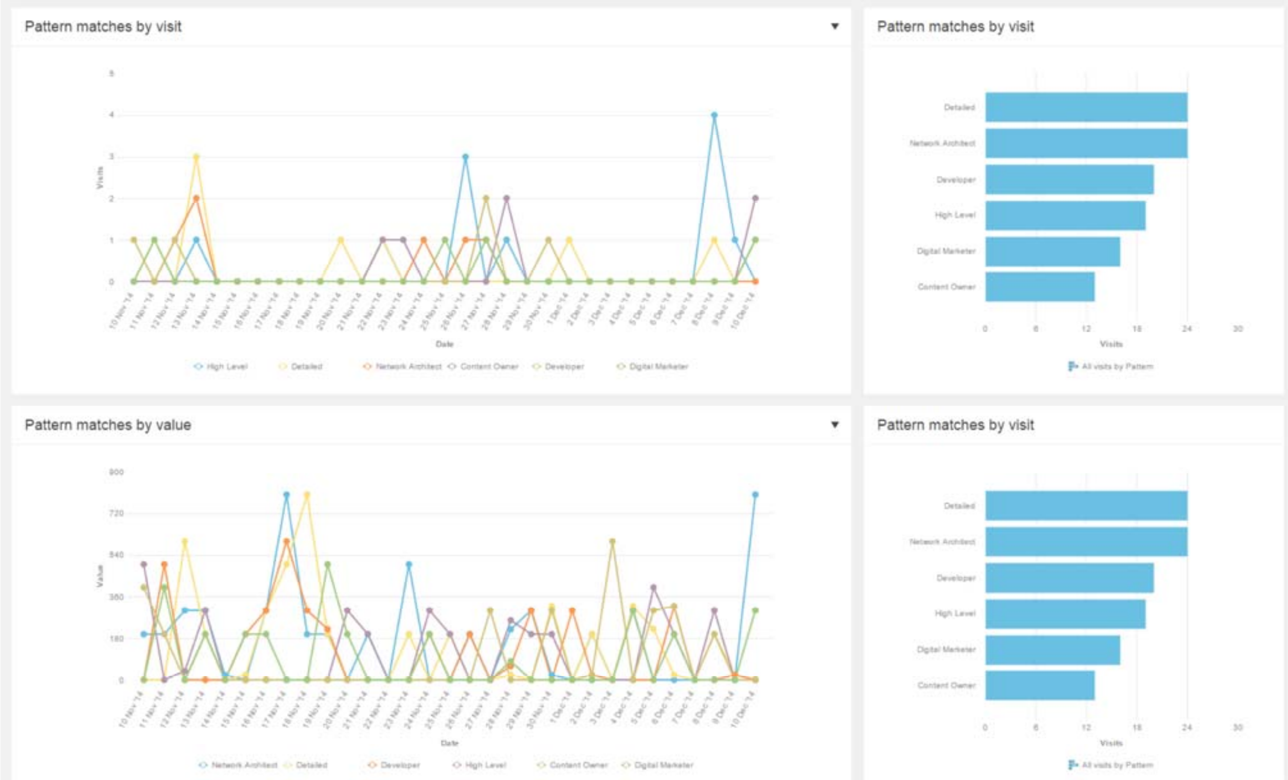
Charts	Description
Online interactions by visits and value per visit	Shows the number of visits within a selected period and the average engagement value generated per visit
Channels groups by visit	Illustrates how much traffic has come to your website and how much traffic each channel has generated
Top campaigns by value	Shows you which campaigns have accrued the highest number of engagement value points
Top pattern matches by value per visit	Shows which pattern cards that contacted have matched provided the highest engagement value per visit
Top goals by conversions	Shows you which goals have been converted the most



The Audience tab

On the Audience tab, you can find the Pattern matches report.

The Pattern Matches report provides you with information about the most common pattern matches and the pattern matches that provide the highest value per visit. There is also a detail report where you can view information about the number of visits, value per visit, average duration, bounce rate, conversion rate and page views per visit for each pattern card.

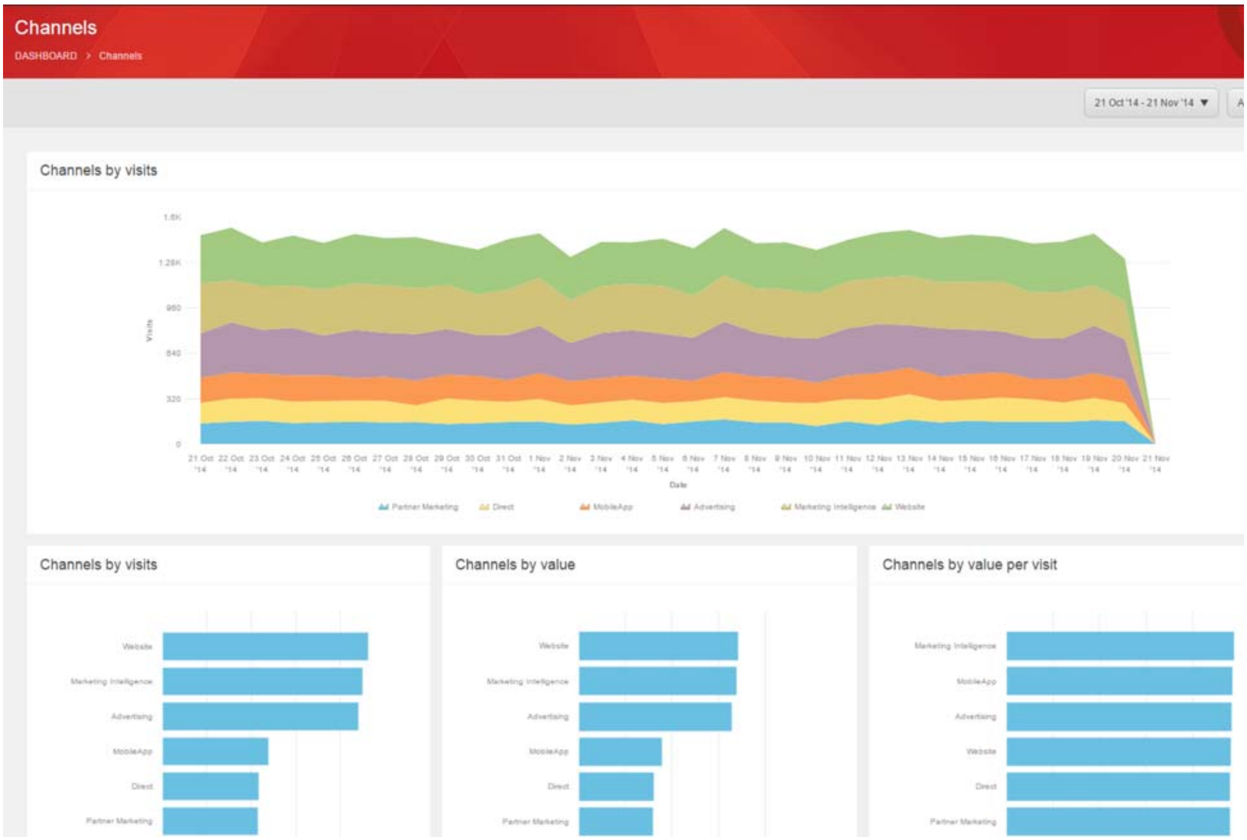


The Acquisitions tab

On the Acquisitions tab, you can find the following relevant charts and dashboards:

- Acquisition Overview – contains five reports that cover high-level information about channels, campaigns, and keywords that contacts use to find your website.

- The Channels dashboard provides you with information about channel traffic by visits. You can also view charts that organize your website's channels by value and by value per visit.



- The Campaigns dashboard shows you graphs and reports that show you your top campaigns by visits and by value per visit.
- You can also view a detail report about your campaigns and campaign groups, which contains information about the value per visit, average duration, bounce rate, conversion rate and page views per visit for your campaigns.

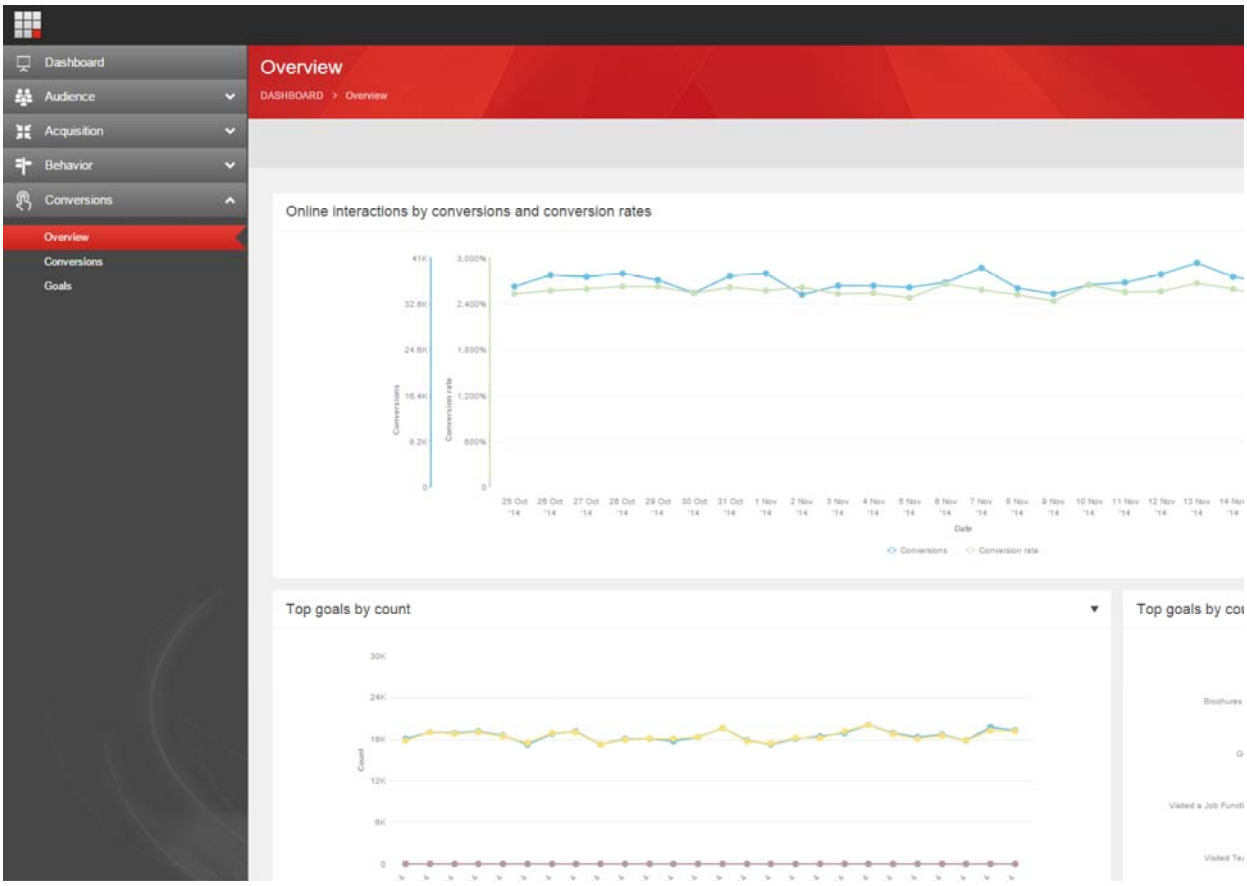
Key	Visits	Value per visit	Average duration	Bounce rate	Conversion rate	Pageviews per visit
Twitter Goal Messages	450	4.5	55s	30%	40%	3.2
Linkedin Share Buttons	432	2.7	50s	22%	22%	3.89
Google+ Share Buttons	421	3.4	2min 4s	14%	28%	3.57
Twitter Share Buttons	321	2.0	2min 15s	33%	17%	4
Twitter Content Messages	302	1.9	41s	33%	16%	3.5
Social Marketer	267	2.7	2min 19s	0%	40%	4
Facebook Share Buttons	231	4.7	2min 5s	0%	25%	5.75
Facebook Goal Messages	154	5.7	18s	50%	75%	2
Facebook Content Messages	84	10.5	2min 17s	0%	75%	4.75

- The Referring sites dashboard provides you with information about the top referring sites by visit and the top referring sites by value per visit. There is also a referring sites detail report.

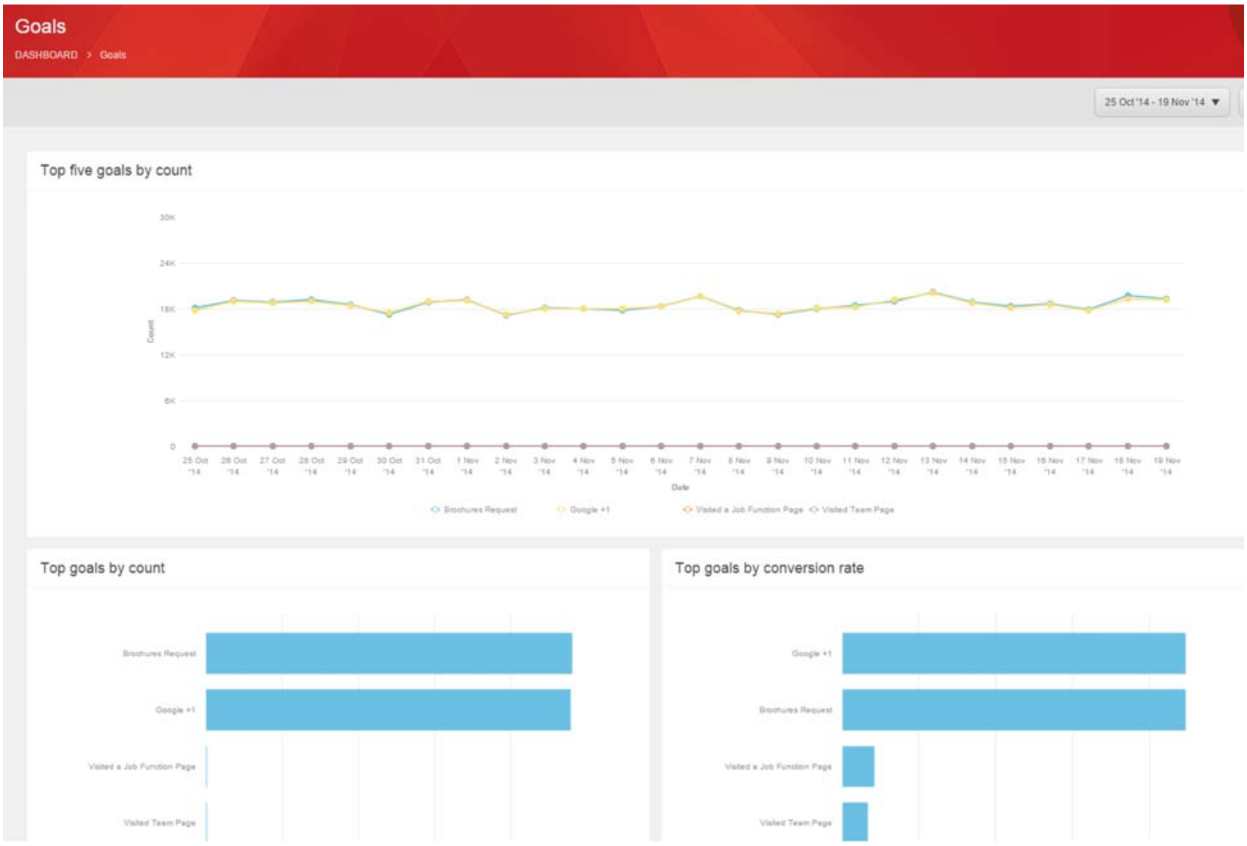
The Conversions tab

The Conversions tab contains the following charts and dashboards:

- The Conversions Overview provides you with information about online interactions by conversions and conversion rates. It also provides a chart and graph showing the top goals by count.



- Under Conversions, the Goals reports focus on the goals that have the highest number of conversions. You can also view the goals that have the highest conversion rates.

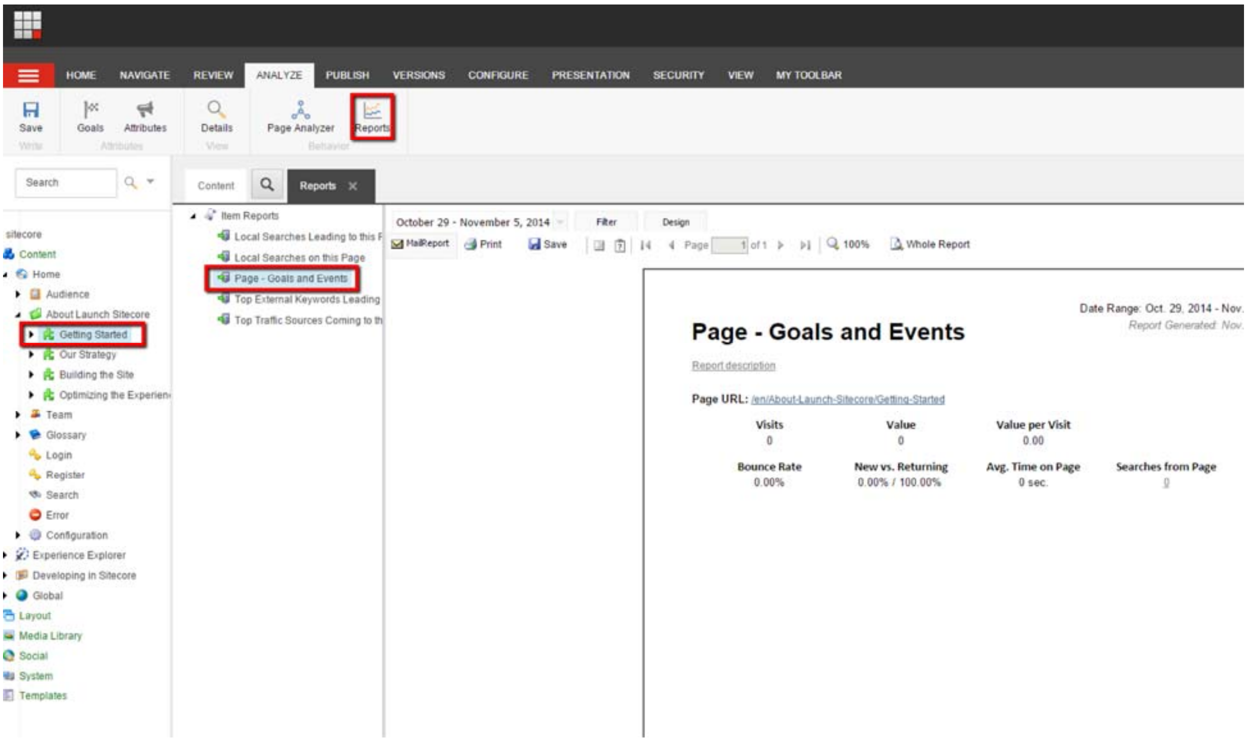


View an item report

Item reports provide analytics information on the performance of individual items or components on your website. For example, you can see the goals and page events triggered by visitors to a particular page.

To view an item report:

1. In the Sitecore Desktop, open the Content Editor.
2. In the Content Editor, in the content tree, select an item.
3. On the Sitecore ribbon, select the Analyze tab and then click Reports.
4. A new Reports tab appears next to the Content tab, to the right of the content tree.



The default item reports that are available from the Sitecore Content Editor are:

- *Local Searches Leading to this Page* - this report lists the search terms that visitors use on your website and which lead them to this page.
- *Local Searches on this Page* - this report lists the words and phrases that visitors searched for while on this page.
- *Page - Goals and Events* - this report shows the value per visit and goal conversion on the page
- *Top External Keywords Leading to this Page* - this report shows the top external keywords visitors used in this time period to come to this page. Keywords are ranked by the value accumulated by visitors who used this keyword.
- *Top Traffic Sources Coming to this Page* - this report lists the top external traffic sources for this page during the selected time period.

Send feedback about the documentation to docsite@sitecore.net.

Walkthrough: Adding a bar chart to an Experience Analytics report

Sitecore Experience Analytics provides dashboards and reports for marketers and marketing analysts to identify patterns and trends in experience data. This data is collected from websites and other external data sources.

You can add bar, line, list, or area chart components to reports using Sitecore Rocks depending on your business needs. For example, you can add a *Conversions by country* bar chart to the *Audience/Overview* report to display how much engagement value is generated on your website by country.

This walkthrough describes all the steps to add a bar chart component to an experience analytics report:

- Add a chart component to a report
- Add a parameters item to the report
- Bind the bar chart control to the parameters item
- Change the chart title

Note

To add any chart component to a report, you must use Microsoft Visual Studio, Sitecore Rocks, and access to a Sitecore website. In Sitecore Rocks, you must also configure a connection to your website.

Add a chart component to a report

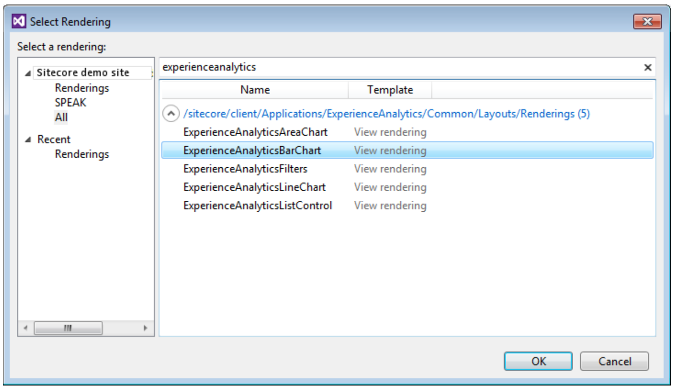
Before you can add a chart component to a report, you first need a report page.

To add a chart component to a report page:

1. In Sitecore Rocks, right-click a report page, click Tasks and then click Design Layout.

2. In the Design Layout window, click Add Rendering.
3. In the Select Rendering dialog box, select a chart control to use in your report page.

   To find all available chart controls, enter a search term beginning with *experienceanalytics*. You can choose from an area chart, bar chart, list, or line chart. For example, to display a bar chart, select the *ExperienceAnalyticsBarChart* rendering.



4. Choose a chart type and click OK to save your changes.

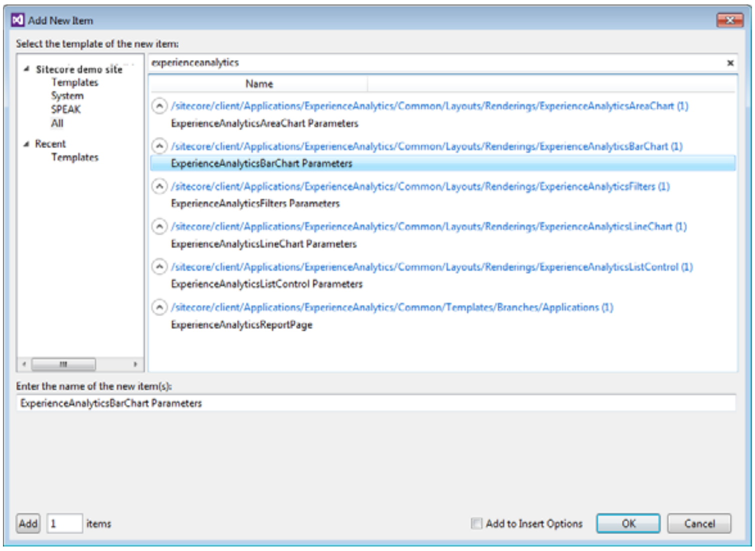
Add a parameters item to the report

You need a parameters item to configure report metrics and other settings in your chart. A parameters item is a subitem of the `PageSettings` item.

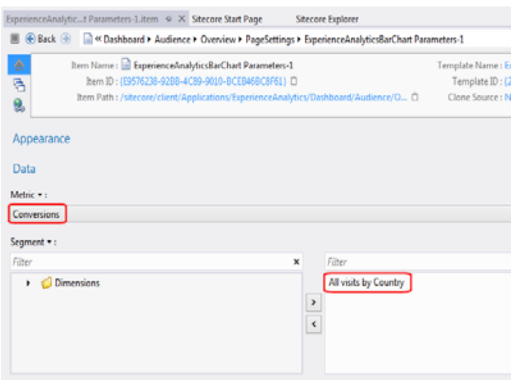
To add a parameters item:

1. On the report page, right-click the PageSettings item and click Add - New Item.

   To find all available chart parameter controls, enter a search term beginning with *experienceanalytics*. You can see parameters items for each type of chart control. For example, if you are using the bar chart control, select the *ExperienceAnalyticsBarChart Parameters* item.



2. You can enter a name for your parameters item or accept the default name, then click OK.
3. To configure the parameters item, double click the item to open it.
4. In the parameters item, select a metric and add a segment to your chart.



5. For example, to show conversions from all visits by country, select the following:
   - Metric = *Conversions*
   - Segment = *All visits by Country*

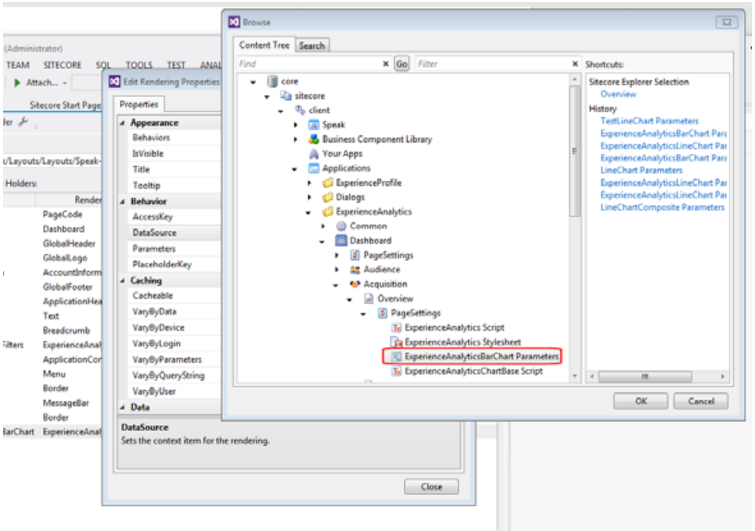
6. Save your changes.

Bind the bar chart control to the parameters item

If you do not bind the bar chart control to the parameters item, your report will not display any data.

To bind the *ExperienceAnalyticsBarChart* control to the parameters item:

1. Right-click the report, click Task, Design Layout, and then double-click the chart item that you have created. For example, double click the *ExperienceAnalyticsBarChart* control.
2. In the Edit Rendering Properties dialog box, in the Behavior section, in Data Source, browse to the location of the *ExperienceAnalyticsBarChart Parameters* item in the content tree.



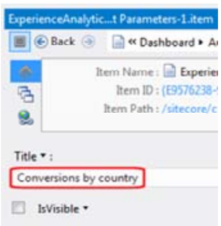
3. Select the parameters item and click OK.
4. Save your changes.

Change the chart title

Your new chart needs a heading to describe the content that it displays.

To change a title label on a chart:

1. Double-click the parameters item for your chart.
2. In the parameters item, expand the Appearance section.
3. In the Title field, enter the relevant title, for example, *Conversions by country*.



4. Save your changes and view the chart in Experience Analytics.

Send feedback about the documentation to docsite@sitecore.net.

Walkthrough: Analyzing campaign efficiency

Sitecore Experience Analytics provides dashboards and reports for marketers and marketing analysts to identify patterns and trends in experience data. This data is collected from websites and other external data sources.

You can use the Campaign Creator to view, filter, and search for specific campaign activities. Once you have selected an individual campaign activity, you can click on a link in the Campaign Creator to go directly to Sitecore Experience Analytics (xAnalytics).

xAnalytics provides you with dashboards and reports that provide you with information about how an individual campaign activity is performing. You can then compare how that campaign activity is performing with other activities within the campaign group, or across your website.

For example, a marketer would use xAnalytics to find out which campaigns provide the highest contact engagement value levels. They would use this data to analyze campaign efficiency, seeing which campaigns provide the highest engagement values per visit. The campaign manager would also compare incoming campaign traffic across channels, showing which channels are the most effective at bringing contacts to campaigns.

This walkthrough outlines how to:

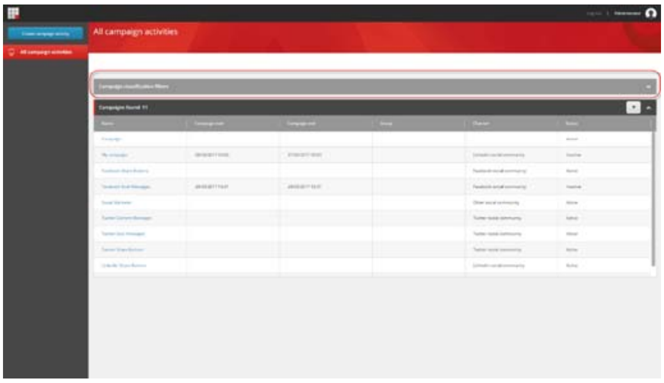
- Select a campaign activity

- [View campaign reports]
- [The Campaign Detail report]
- [The Channels reports]

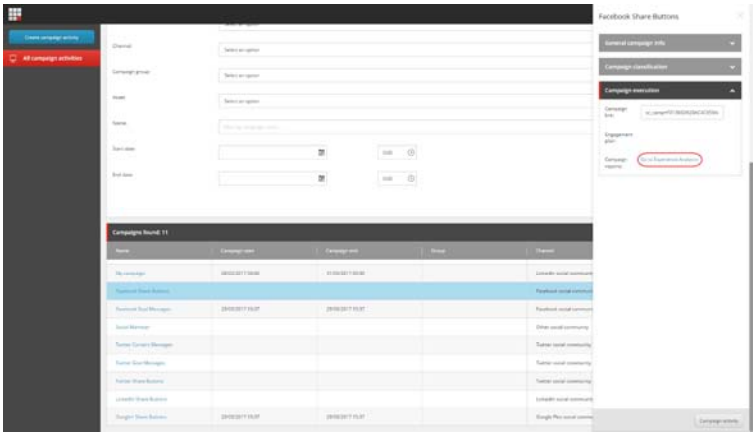
Select a campaign activity

Use the Campaign Creator to find the campaign activity you want to analyze.

1. In the Sitecore Launchpad, open the Campaign Creator.
2. On the All Campaigns page, under *Campaign classification filters*, open a drop-down menu to find and select a filter that you want to apply to the campaign activities.



3. Click the Show Campaigns button to apply your filter settings. You can find all the campaigns that match your filter criteria in the Campaigns found section.
4. In the Campaigns found section, click the relevant campaign activity. Then, to open xAnalytics, in the campaign activity panel, under Campaign execution, click Go to Experience Analytics.

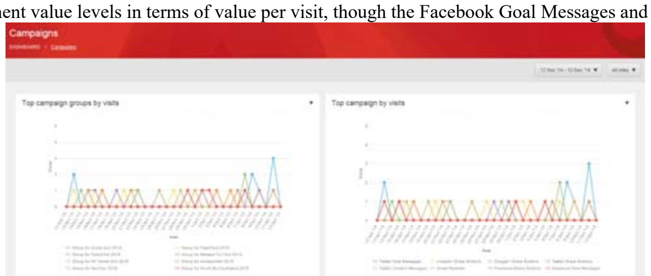


View campaign reports

In the Experience Analytics dashboard, click the Acquisition tab and then click the Campaigns subtab. This opens up the *Campaigns Overview* report. In the *Campaigns Overview* report, you can view charts and graphs which show you which campaigns have brought the highest number of visits to your website.

You can also view data on which campaigns have the highest engagement values per visit.

In the following example, the campaigns are providing similar engagement value levels in terms of value per visit, though the Facebook Goal Messages and the Twitter Goal Messages have more visits than the other campaigns have.



The Campaign Detail Report

In the Experience Analytics dashboard, in the Acquisitions tab, in the Campaigns subtab, you can view the *Campaign Detail* report. This is a table list located at the bottom of the *Campaigns Overview* report.

The *Campaign Detail* report provides you with the following information:

Campaign detail column	Comments
Visits	The total number of visits to the campaign.

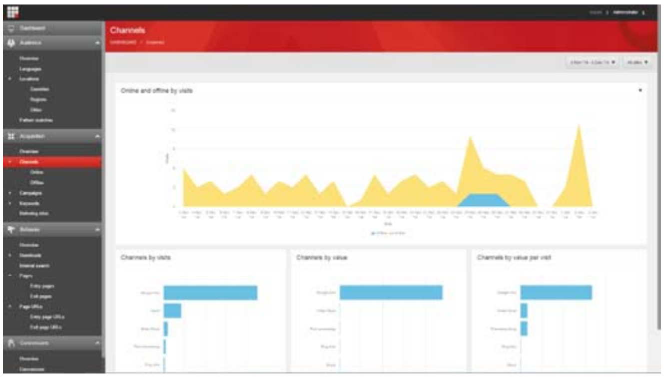
Value per visit	The average engagement value per visit to the campaign.
Average duration	The average length of a contact's visit to your website.
Bounce rate	The bounce rate.
Conversion rate	The goal conversion rate for visitors to the campaign.
Page views per visit	The average number of pages that contacts visited.

In the following example, you can see that Twitter Goal Messages have the highest number of visits overall. Facebook Goal Messages and Facebook Content Messages, however, have higher goal conversion rates despite having fewer visits overall.

Key	Visits	Value per visit	Average duration	Bounce rate	Conversion rate	Pageviews per visit
Twitter Goal Messages	459	4.5	55s	30%	40%	3.2
LinkedIn Share Buttons	432	2.7	58s	22%	22%	3.89
Google+ Share Buttons	421	3.4	2min 4s	14%	28%	3.57
Twitter Share Buttons	321	2.8	2min 15s	33%	17%	4
Twitter Content Messages	302	1.8	41s	33%	16%	3.5
Social Marketer	267	3.7	3min 15s	0%	40%	4
Facebook Share Buttons	231	4.7	2min 5s	0%	25%	5.75
Facebook Goal Messages	154	5.7	18s	50%	75%	2
Facebook Content Messages	84	10.5	2min 17s	0%	75%	4.75

The Channels reports

In the Experience Analytics dashboard, in the Acquisitions tab, click the Channels subtab to view the *Channels Overview* report.

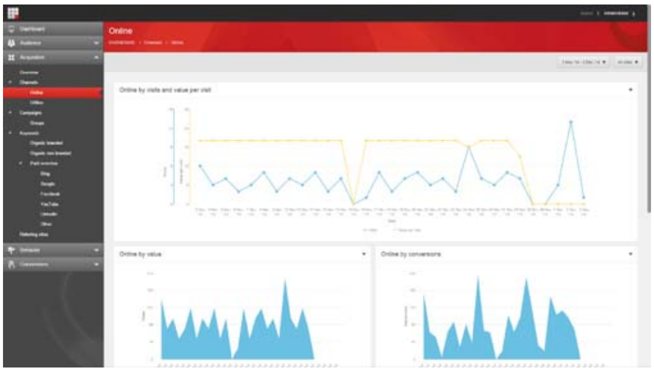


The Channels overview provides the following reports:

Report name	Comments
Online and offline by visits	The total number of online and offline interactions that contacts had with your website.
Channels by visit	The top performing online and offline channels by number of visits.
Channels by value	The online and offline channels which provided the highest contact engagement value levels.
Channels by value per visit	The most effective online and offline channels by contact engagement value generated per visit.

In this example, you can see that Google Ads receives the highest traffic, and provides the most engagement value per visit.

In the Channels subtab, click Online to view the *Online Overview* report about incoming traffic coming exclusively from online channels.



You can see, for example, in the Online by value and Online by Conversions reports, that the engagement values and number of goal conversions vary from day to day.

The Online by visits and value per visits report shows, however, that the value per visits level remained reasonably constant despite fluctuations in traffic levels.

Send feedback about the documentation to docsite@sitecore.net.

Content testing overview

Content tests let you test alternative variants of your website's content to find out which pages, components, or combinations of components are the most effective with visitors. The content or components that you are comparing in a test should be designed to achieve the same goal. You can quickly evaluate the test results and gain insight into the content that functions best with your audience. This can help you create more successful campaigns with higher goal-conversion rates.

You can use the following tests for testing content in Sitecore:

- Basic content tests
- Component tests
- Page tests

You can let the tests run for a suggested amount of time to generate statistically significant data and then access the test results for a page on the Optimization tab of the Experience Editor. When you have finished testing, you can compare conversion rates and select the page or component that you would like to use on your site.

Basic content tests

When you edit a page in the Experience Editor, Sitecore creates a new version of the page.

Note

Users with admin rights do not need to lock a page to edit it. Therefore, users with admin rights must create a new version manually.

After you save and submit the page, Sitecore proposes a test to compare the effect of the old and new versions of the page. You can preview the test, edit the test parameters, and start the test if you want.

You can create basic content tests in both the Content Editor and the Experience Editor. The Experience Editor is the more convenient and flexible tool to use because it lets you use existing content and create new content without leaving your website.

Component tests

You can create variants of components with different content, design, and personalization rules. Creating component variants lets you create complex tests with which you can:

- Test more than two variants of a single component against each other.
- Test multiple components at the same time, and each version of a component can be tested in combination with each version of the other components.

You can create variants of components in both the Content Editor and the Experience Editor, but the Experience Editor is the more convenient and flexible tool to use because it lets you use existing content and create new content without leaving your website.

You define alternative website components either by creating new content or by cloning and altering existing content. If you assign personalization rules to components, the personalized components are included in a component test as variants of the original components.

When you submit your changes, Sitecore proposes a test to compare the effect of the original and the new variants. You can preview the test, edit the test parameters, and start the test if you want.

Page tests

A page test tests a complete page against another page. The pages in the test can be different versions of the same page, or they can be completely different pages. This form of testing is useful for, for example, landing pages.

You can create and run page tests in Experience Optimization.

Send feedback about the documentation to docsite@sitecore.net.

Cancel active tests

Content tests normally stop automatically after enough data has been gathered for a winner to be picked. However, you might want to manually cancel active content tests. For example, before upgrading your Sitecore solution, you must cancel all active content tests.

To stop active tests:

1. In the Experience Optimization application, open the Active Tests list.

   Optimiz" height="141" width="519" alt="Picture 1" title="" class="documentimage" doc-fancy-image="">

2. In the list, click a test that you want to cancel. The Experience Editor opens.
3. On the Optimization tab, in the Page Reports group, click Test result:

   Can" height="127" width="522" alt="Picture 2" title="" class="documentimage" doc-fancy-image="">

4. In the Test result dialog box, to use the original experience, click Cancel Test. To use another experience, select an experience in the Top experiences list and click Pick as winner.

   Picture 3

5. Return to the Experience Optimization application and repeat steps 1-4 to cancel as many tests as you want.

Send feedback about the documentation to docsite@sitecore.net.

Personalization tests

After you apply personalization rules to the variants of a component, you can test the effect of the personalization. By default, all tests in Sitecore use the Subgroup Round Robin Sticky test strategy. This topic describes how personalization tests and the default test strategy work so you can understand what contacts will see when they visit your site.

This topic contains the following sections:

Evaluating the effect of content on a specific group

In order to evaluate the effect of new content on a specific target group, you need to compare it to the effect of the default content on the same target group.

During a simple A/B content test, half of the contacts see the original content and half see the new version.

In a personalization test:

- All contacts who do *not* meet the personalization criteria see the original content.
- Of the contacts that *do* meet the personalization rule criteria, *half* see the new variant and *half* see the original content.

Example of a test with one personalization condition

A page that you test has two versions – the original (default) version, and a version with the condition "When the current day is Sunday".

- All contacts who visit the site from Monday through Saturday see the default version of the variant.
- Half of the contacts who visit the site on Sunday see the personalized version of the content and the other half see the default version.

Testing multiple personalization conditions

A personalization test can include multiple conditions applied to the same component. In that case, a round robin strategy is used to ensure that all conditions that apply to a group of contacts are presented with equal frequency to that group.

Example of a personalization test with multiple conditions

The following table shows the conditions applied to a personalized component with two conditions besides the default condition that is being tested:

Condition Name	Rule	Experience

Condition 2	When the current day is Monday	Experience 2
Condition 1	When the current day of the month is equal to 1	Experience 1
Default	Always applies	Default

In the Personalize the component dialog, the conditions look like this:


Picture 1

Contacts who visit the website belong to one of four groups based on which conditions they meet. The following table shows the four groups, and describes which experiences are presented to each group:

Group	Description	Matching conditions	Experience shown to contacts from the group
Group 1	Contacts who visit the site when the day is not Monday, and it is not the first	Condition 2 = False Condition 1 = False Default = True	All contacts in this group see the default
Group 2	Contacts who visit the site when the day is Monday, and it is not the first	Condition 2 = True Condition 1 = False Default = True	Contact 1 - Experience 2 Contact 2 - Default Contact 3 - Experience 2 Contact 4 - Default Contact 5 - Experience 2
Group 3	Contacts who visit the site when the day is not Monday, and it is the first	Condition 2 = False Condition 1 = True Default = True	Contact 1 - Experience 1 Contact 2 - Default Contact 3 - Experience 1 Contact 4 - Default Contact 5 - Experience 1
Group 4	Contacts who visit the site when it is Monday, and it is the first	Condition 2 = True Condition 1 = True Default = True	Contact 1 - Experience 2 Contact 2 - Experience 1 Contact 3 - Default Contact 4 - Experience 2 Contact 5 - Experience 1

As illustrated by the table, contacts from each group see the experiences that apply to that group, including the default experience. For example, group 4 comprises contacts who visit the site on the first of the month, when it is also Monday. Since these contacts meet both condition 1 and condition 2, one third of these contacts see Experience 1, one third see Experience 2, and one third see the default experience.

Testing a page with multiple personalized components

You can run a personalization test on a page that contains multiple components with personalization conditions. The same principles apply as with the simpler tests; all conditions that apply to a group of contacts, including the default condition, are presented with equal frequency to that group.

If, for example, there are two components that have two conditions in addition to the default, then there is a total of 9 experiences. The more experiences that there are in a test, the larger the sample of contacts that is required to run the test in order for the results to be statistically significant/valid.

Testing a component with already tested personalization rules

Sometimes you may want to add new personalization rules to a component that has one or more rules that have already been tested. An already tested rule will not be tested again. When a contact views an experience that was already tested, the test counts the visit as if the contact viewed the default condition.

Send feedback about the documentation to docsite@sitecore.net.

The Personalized experience page

The Personalized experience page shows the statistics for any personalization rules that are implemented on a page as well as statistics about any personalization rules that are being tested on the page. To open the page, in the Experience Editor, on the Optimization tab, click Personalization. Sitecore only enables the Personalization button when there are active personalization rules on the page.

Personalized_experience

The dialog shows the following:

- A summary of the number of rules and components on the page (for example, *6 personalization rules in 2 components on this page*).
- A list of all rules, grouped by component (for example, *Promo with image* and *SmallStickyNote*). The graphs show the experience effect of each rule. The effect is the trailing value/visit for the personalized variation calculated against the not personalized variation.
- A preview of the page as it looks when the currently selected rule is in effect. The relevant component is highlighted. The other components use their default rules.
- The name and description of the currently selected rule.
- The Trailing Value/Visit of the currently selected rule, which is calculated as the total trailing engagement value per visit, counting only page views occurring after the visitors encountered the page being tested. It shows the value for the personalized and not personalized variations of the component.

  The Personalized experience page only shows the Trailing Value/Visit data for components that are being tested.

- The Effect of the personalized variation, which is calculated as the difference in percent between the trailing values/visit of the personalized and the not personalized variations.
- The Reach of the currently selected rule, which is the number of visitors exposed to the rule and what percentage this is of all visitors.

The Edit personalization button (you have to scroll down to see it) lets you edit the currently selected personalization rule directly from the dialog box.

Send feedback about the documentation to docsite@sitecore.net.

Test lists

You can see test lists in both the Experience Editor and in the Experience Optimization application. You use the test lists to browse lists of draft page tests, active tests, historical tests, and suggested tests, and to navigate to these tests.

In the Experience Editor, on the Optimization tab, you can see the tests that you have created.

Users with access rights to the Experience Optimization application can see lists of tests created by all users. The test lists are in the Lists part of the menu.

This topic describes the following:

- Active tests
- Historical tests
- Suggested tests
- Draft tests

Active tests

This is a list of active tests. The following information is available:

- Page – the page name.
- Experiences – the number of experiences.
- Start date of the test – the date that the test was started.
- Days left – the number of days left to complete the test.
- Created by – the name of the user who created the test.
- Effect – the current effect of the test.

Historical tests

This is a list of finished tests. The following information is available:

- Page – the page name.
- Experiences – the number of experiences.
- The end date of the test – the date that the test ended.
- Test duration – the number of days that the test ran.
- Created by – the name of the user who created the test.
- Test score – the score that the test achieved.
- Effect – the effect of the test.

Suggested tests

This is a list of suggested tests. Sitecore continually analyzes the tests run on the website and the usage of the website in general, and uses this analysis to suggest new tests to run. (A system administrator can specify which content areas are included in the analysis.)

Note

All scores are normalized between 0 and 100.

The following information is available:

- Page – the page that Sitecore suggests testing.
- Potential – shows which pages have a high potential for increasing the engagement for the users visiting the page.

  The potential is calculated as the value that the page accrues compared to the site, relative to how often the page is viewed. It is calculated as follows:

  Potential = ("item views" / "site views") - ("item value" / "site value")

- Impact – the expected impact of the test.

  The impact is calculated based on how often the page is viewed compared to all pages on the site. It is calculated as follows:

  Impact = "item views" / "site views"

- Recommendation – a combination of potential and impact. This is the key score, and indicates how highly Sitecore recommends optimizing this page.

  The recommendation is calculated as follows:

  Recommendation = ("potential" + "impact") / 2

Draft tests

This is a list of page tests that have been created and saved but have not been run yet.

The following information is available for each test:

- Page – the name of the page.
- Experiences – the number of experiences.
- Created by – the name of the user who created the test.
- Last saved – the date when the test was last saved.

Note

The Draft tests list is only available in the Experience Optimization application.

Send feedback about the documentation to docsite@sitecore.net.

Content test results

Sitecore sends an email to content authors when a test that they have started is over. The email contains a link that you can click to see test results. You can also navigate to the page in the Experience Editor and open the test results from the Optimization tab.

Opt" height="33" width="454" alt="optimization" title="Optimization" class="documentimage" doc-fancy-image="">

A red dot on the tab indicates that there is an active test on the page. The ribbon of the Optimization tab looks like this:

ee-tab

If there is an active test, the Status shows how many visitors are required for the test to finish. In this example, the status shows that it will take another 3140 visitors to finish the test. If there is enough historical data, the status instead shows the number of days it will take for the test to finish.

Test winners

This topic describes:

- Test winners
- Test results
- Engagement value
- Conversion rate
- Variables
- Top goals converted
- Top clicks to pages
- Site usage

Test winners

The user who has created a test can select an experience and click Pick as winner to end the test and declare the experience as the winner of the test. This overrules the test objective defined for the test. Otherwise, Sitecore finds a winner.

When a test winner has been found, the Test summary window is displayed.

test-summary

As part of the results, you can see:

- Other pages that you can optimize and test
- Predictions of specific visitor segments that prefer a different experience than the test winner.

In this example, the test has identified that visitors arriving during workdays prefer the original experience instead of the winner. You may want to create specific personalization rules that target these visitors, and then run a test with the new rules.

Test results

You can click the Test result button in the Optimization tab to see the current test result report for the page.

Picture 14

The Overview shows:

- Best experience effect – the performance of the best performing experience, calculated as the difference in percent between the trailing value per visit of the original and the best experience in the test.
- Confidence level – the statistical confidence level that the test has reached. If there are fewer than 100 visits, the confidence level is 0.
- Score – the test score assigned to the creator of the test, based on the engagement value gained and the importance of the tested page.

The Engagement Value, Conversion Rate, and Variables tabs show bar charts with the tested experiences. When you select one of the experiences, the Details panel shows details about this experience.

Below the Details panel there are three expandable sections with more information about the currently selected experience.

Engagement value

This tab contains a graph that lists the experiences in the test in order of their performance, as measured in terms of engagement value. The best performing experience is assigned a value of 100, and the bars in the chart indicate the relative performance of the remaining experiences. The bar is green for experiences that perform better than the original experience; the bar is red for experiences that perform worse than the original.

Select an experience to see this information in the Details panel:

Trailing value/visit " height="250" width="370" alt="details" title="Details" class="documentimage" doc-fancy-image="">

There are two general measurements for the selected experience:

- Trailing value/visit – the total engagement value, counting only page views occurring *after* the visitors encountered the page being tested, divided by the number of visits exposed to the experience.
- Experience effect – the total negative, positive, or neutral experience effect. This is the difference in percent between the trailing value/visit of the selected experience and the default experience.

The Details panel also shows a preview of the experience. You can click Preview to edit the selected experience.

Conversion rate

This tab shows the Conversion rate for all the experiences. Select a goal from the Goal dropdown list to see how many times the goal was triggered. The conversion rate is expressed as a percentage of the visits in which the selected experience was displayed. If there is more than one conversion per visit, the rate is greater than 100%.

The bar chart shows statistics for the experiences in the test.

When you select an experience in the bar chart, the Details panel shows the conversion rate percentages for the selected experience in addition to the trailing value per visit and the experience effect.

Variables

The Variables tab shows a graph indicating the engagement value of each variable in the test.

The Details panel shows the trailing value per visit and the experience effect.

Top goals converted

Expand the Top Goals Converted section to see the goal conversion results for the selected experience.

Goal – the goals tr" height="234" width="325" alt="top-goals-converted" title="Top Goals Converted" class="documentimage" doc-fancy-image="">

The table shows the following metrics:

- Goal – the goals triggered by visitors exposed to the test
- Conversions – the number of conversions
- Rate – the number of goal conversions after viewing the selected experience divided by the number of visits, multiplied by 100
- Change – the change between the original experience and the selected experience

Top clicks to pages

Expand the Top Clicks to Pages section to see the top clicks for the selected experience.

top-clicks

The table shows these metrics:

- Page – the name of the next page that was clicked in the session
- Clicks – the total number of clicks to the page
- Rate – the number of clicks to page after viewing the selected experience divided by the number of visits, multiplied by 100
- Change – the change between the original experience and the selected experience

Site usage

Expand the Site Usage section to see a summary of the site usage for the selected experience.

site-usage

The table shows the following metrics:

- Type defines the value in the second column:
  - Bounce rate (percent) – the percentage of visitors who visit this page as their first page and then bounce (leave the site) without viewing other pages.
  - Avg. duration on page (number in seconds/minutes) – the average duration that visitors stay on the page
  - Avg. duration for visit (number in seconds/minutes) – the average duration that visitors stay on the site
  - Pageviews per visit (number of pageviews) – the average number of different pages that visitors look at in one visit
- Change shows the change of this measurement compared to the original.

Reach

This shows the number and the percentage of visitors exposed to this experience.


reach

Note

As described above, some results are calculated based on number of visits to the page being tested and this number can differ from the number of visitors, since a visitor can have multiple visits to the same page. The number of visits is not shown in the Test Result dialog.

Send feedback about the documentation to docsite@sitecore.net.

Using the Dashboard

The dashboard is part of the Experience Optimization application. The dashboard shows a number of key performance indicators.

What Different Roles Can See

What you can see in the dashboard depends on the role that the administrator has assigned to you.

You need to have the "Analytics Testing," "Analytics Advanced Testing," or the "Analytics Management Reporting" role to see the Experience Optimization application at all.

If you have the "Analytics Testing" or "Analytics Advanced Testing" role, you can see:

- Overview of current activity
- Leaderboards

If you have the "Analytics Management Reporting" role, you can see:

- Overall performance for the current month compared with the previous month
- Leaderboards

Some things are unique for the roles, and some are common for all roles (but display different data, depending on the role.)

Overall Performance

Overall Performance shows the combined performance changes between the current month and the previous month for *all* content authors.

- Score – calculated as the product of the effect and number of visits in a month
- Guessing – the percentage of correct guesses about the outcome of tests
- Effect – the difference between the trailing values of the best non-original experience and the original experience
- Activity – the number of tests run

Experience Optimization shows the changes between the two months as percentages.

Leaderboards

The front page of the dashboard shows a number of leaderboards. The dashboard shows each leaderboard twice: one version with data from the current month and another with data from the previous month. They show users that have started tests ranked by a number of measures (KPIs.)

The leaderboards show the scores as bar charts, and these bar charts are normalized.

The leaderboards are:

- The main leaderboard – shows a combined score, and scores for guessing, effect, and activity. It sorts users by their test score. Overall Performance describes how these scores are calculated
- Best Guessing – shows scores for guessing
- Highest Effect – shows scores for effect
- Most Active – shows who has created most tests

Depending on you role you see these leaderboards differently.

Users with the "Analytics Testing" or "Analytics Advanced Testing" roles:

- See their own score as well as the scores of the users ranked just above and just below them. If users have no data for a specific KPI, they will see a "no data" message.

Users with the "Analytics Management Reporting" role:

- See the top scores and the average score.
- Can click View full list for each leaderboard and see more data:
- See the average score as well as the scores of *all* users.
- Can move the period that is shown back and forth in time with the Previous and Next buttons.

- Can change the "time resolution"

The views you get to after clicking View full list are reports, and you can get to these reports in the Leaderboard group in the Reports menu.

Current Activity

"Current Activity" shows the recent activity of a content author. This is what content authors see at the top of the dashboard when they open the Experience Optimization application:

- The number of tests started this day (by all users)
- The number of tests currently running
- The number of visitors that exposed to tests this day (this is not the same as the total number of visitors: the percentage of visitors exposed to the test can be set for each test.)
- The overall outcome of tests that ended this day.

Send feedback about the documentation to docsite@sitecore.net.

Adding content testing to a workflow

The main entry point into content testing for users is the *workflow*. Developers can specify how content testing is used in the workflows that they create.

The following diagram shows the *Sample Workflow* in Sitecore, highlighting the relevant parts for content testing.

Name

Workflow part	Name	Purpose
Action	Launch Create Test Dialog	Launches the Create Test Dialog to enable the user to create a test
Command	Approve without Test	Enables the user to bypass the Create Test Dialog for small and minor updates
Action	Remove Tests	Removes any tests that you do not want to run

You can add content testing to a workflow by identifying where in your workflow you want Sitecore to launch the Create Test Dialog. The dialog must be launched from a workflow command:

- Under the workflow command definition item, add a child item and base this item on the *Launch Create Test Dialog Action* template (`/sitecore/templates/System/Content Testing/Workflow/Launch Create Test Dialog Action`). Call this item `Launch Create Test Dialog`.
- Identify an alternate route in your workflow for when the user does not want to create a test for their changes.
  - Sometimes, you need to add a new path between your existing workflow states by adding a new workflow command to make the workflow work. The `Approve without Test` command in the sample workflow is an example of this.
- Add a child item under the workflow command definition item and base this item on the *Remove Tests Action* template (`/sitecore/templates/System/Content Testing/Workflow/Remove Tests Action`). Call this item `Remove Tests`.

UI changes

When you add a workflow command to provide an alternate route, you sometimes have to change the UI text of the workflow commands. The workflow command names often indicate whether the test is being created or not, for example:

- Approve with Test
- Approve without Test

For usability reasons, you should change the UI when there are no test candidates in the test creation dialog. You can hide the workflow command entirely, or you can change the name.

Hiding the command

For better usability, hide the command that launches the test creation dialog when there are no test candidates for the item.

To do this, you change the `Appearance Evaluator Type` field of the command definition item to: `Sitecore.ContentTesting.Workflows.TestCandidatesCommandStateEvaluator, Sitecore.ContentTesting`. This workflow command appearance evaluator hides the command when there are no test candidates for the item in the workflow.

Changing command names

You must change the name of a command that bypasses the test creation dialog when there are no test candidates for the item.

You do this by changing the `Appearance Evaluator Type` field of the command definition item to: `Sitecore.ContentTesting.Workflows.TestCandidatesCommandNameEvaluator, Sitecore.ContentTesting`.

The workflow command appearance evaluator changes the name of the command to `Approve` if there are no test candidates for the item in workflow.

Send feedback about the documentation to docsite@sitecore.net.

Create and run page tests

Page tests are different from content tests: a page test tests a complete page against another page. The pages in the test can be different versions of the same page, or they can be completely different pages.

You normally start page tests from the Experience Optimization application, although it is also possible to start a page test from the Experience Editor.

In order to create and run a page test, you must:

- Select pages to test
- Review selected pages
- Start the test

After the test, you can:

- View test reports

Select pages to test

To select the pages to test:

1. Open the Sitecore Launch Pad, click Experience Optimization, and then click Create Page Test.
2. Select the original page to test. You can navigate to a page, or you can search for it.
3. Click the plus icon to select pages to test against:

   pages-to-test-against

   In the Add page dialog, select one of the following options:

   - use existing: select an existing page to test against.
   - previous page version: select the previous version of the current page (if one exists).
   - new page version: create a new version of the current page. You can edit the new version when you review the test.

   Picture 17

4. Click the plus icon again to select more pages until you have selected all the pages that you want to test.

Review selected pages

When you have selected the pages that you want to test, you can review the test. Sitecore displays information about the test, and you can adjust a number of parameters for the test.

Sitecore displays previews of the pages that you have selected, with one variation selected and the other variations in the carousel under the selected variation. To select a different one, click any variation in the carousel.

You can see the following information:

- An information bar that tells you how long the test is expected to need to run before it can declare a winner

  Picture 13

- Information about the currently selected variation
- A test summary

You can adjust the following parameters:

- My expected effect of changes: make a guess about the change in engagement value of the changes you have made.
- Percentage of visitors exposed to test: Specify the percentage of visitors that you want exposed to the test.

For example, if you set the percentage at 40%, 40% of visitors will see one of the variations that you have created. The original variation is part of these. The remaining visitors – 60% in this example – will see the original variation.

Sitecore calculates test results based on the visitors it exposes to the test.

- Statistics: Select the statistical confidence level that the test needs to reach before it can declare a winner. You select one of 90%, 95%, or 99%. The higher the level, the longer the test needs to run.
- Test objective: Select a test objective. You select either one of the goals specified in the Marketing Control Center, or you select *Trailing Value/Visit* (the engagement value based on page views occurring after the visitors encountered the page being tested, divided by the number of site visits).
- Select how to pick a winner: Select how you want the winner picked:
  - Automatically select a winner based on the test objective
  - Automatically select a winner based on the test objective, unless it significantly decreases engagement value
  - Manually select winner
- Duration: Specify the minimum and maximum time for the test to run:
  - Minimum: select 3, 7, or 14 days.
  - Maximum: select 14, 30, or 00 days.

Start the test

When you are done reviewing and adjusting, start the test:

- Click Start test.

View test reports

When the test has found a winner, you can view the test results in Reports. You can see the test summary as well as:

- An overview that shows the best page effect, the statistical confidence level, and the score gained.
- A bar chart that shows the engagement value for the different pages in the test.
- The conversion rate based on the goal that you select.

Send feedback about the documentation to docsite@sitecore.net.

Create variants of a component

Component tests let you test alternative combinations of your website's content to find out which components or combinations of components are the most effective with visitors. You can quickly evaluate the test results to get insight into the content that functions best with your audience. This can help you create more successful campaigns with higher goal-conversion rates.

When your test is finished, you can see each test variation and the accumulated value next to each one and pick the winning content.

This topic describes how to:

- Create a new variation of a component
- Specify the content for a variation
- Specify the design of a variation
- Save new variations

You can also personalize components and then include the personalized components in a component test.

Create a new variation of a component

To create a new variation of a component on your website:

1. In the Experience Editor, on the Versions tab, select the language version that you want to create a test for.

Picture 13

Important

When you run a component test, you must only run the test in one language version at a time. If you run the same test in multiple languages at the same time, the variation that you first select as a winner is also selected as the winner in all the language versions that you have run the test on.

2. Select the component on your website that you want to test and in the floating tool bar, click Test the Component .
3. In the Test the Component dialog box, to create a new test variation, click New Variation.

cms_EE_testcomponent_newvariation

4. Replace *Variation Name* with a name for the test variation that you are adding.

Now you can specify the content and design of the new variation before you start a content test.

Specify the content for a variation

To specify the content for a new test variation:

1. To browse for existing content in the content tree, in the Test Content Item field, click Browse
2. To b" height="13" width="13" alt="Picture 4" title="" class="documentimage" doc-fancy-image="">.
3. In the Select the Associated Content dialog box:
    - To browse for an item in the Sitecore content tree, click Select Existing Content.
    - To search for an item you want to add to the test, click Search for Content.
    - To clone an existing item, then edit it, click Clone Current Content.
    - To insert a new content item, click Create New Content.

Picture 14

4. When you have selected the content that you want to use, click OK.

Note

Alternatively, to hide the component on the page, select Hide Component.

Specify the design of a variation

To select a rendering for the new variation:

1. In the Test the Component dialog box, to test the design of the component, select the Enable variation of component design check box.
2. In the Test Component Design field, click Browse to select a different rendering.

Picture 23

Save new variations

Once you have added all the variations that you want to include in your test, you must save your changes before you start the test:

1. In the Test the Component dialog box, click OK.
2. Save your changes.

Now you are ready to start the content test.

Send feedback about the documentation to docsite@sitecore.net.

Preview and start a content test

You can create a content test to evaluate alternative versions or combinations of your website's content and find out which pages, components, or combinations of components are the most effective with visitors.

Note

This topic describes how to start a basic content test or a component test. If you want to compare the effects of multiple versions of a page, or two completely different pages (rather than two versions of the same page), you must create and run a page test.

After you edit a page, add personalization rules to components, or create variants of one or more components, you can start a test in the Experience Editor. This topic outlines how to:

- Preview a test
- Adjust test parameters
- Start a test

Preview a test

To preview a test in the Experience Editor:

1. Save and submit the item that you want to test.
2. In the notifications at the top of the page in the Experience Editor:
    - If you are using a workflow, click Approve with Test:

      message-bar

    - If you are not using a workflow and you have enabled test notifications, click Create a test:

      Picture 3

3. Enter a comment and click OK.

   The Preview and start test dialog box opens. On the Preview tab, you can preview the test and predict the outcome:

Picture 7

4. On the Preview tab, you can see the experiences you have created that you can test. Select a different experience from the carousel to see a larger preview of it.
5. To set a filter on the experiences that are displayed in the carousel, click the View all drop-down and select a row in the table to display only one type of variation:

   Picture 9

6. In the My expected effect of changes section, make a guess about the effect that you expect your changes will have:

   Picture 4

   You can select:

   - I expect a negative change in engagement value
   - I expect no significant change in engagement value
   - I expect a positive change in engagement value

   In a test with multiple experiences, you are guessing about all experiences against the original.

   Note

   There are situations where the I expect a negative change in engagement value option is useful. For example, the content author disagrees with a suggestion about a change in the content, and uses this option to start a test to show that they are right about the suggestion.

   Sitecore calculates a score to show how good you are at guessing. You can view the performance reports to see the results and your score.

Adjust test parameters

You can adjust a number of test parameters. However, some options lower the quality of the test result. You must decide which trade-off between quality and time best suits your situation.

To adjust the test parameters:

1. In the Preview and start test dialog box, on the Variables tab, in the Variables in test section, reduce the number of experiences by clearing the check box for one or more variables:

   Picture 6

   Note

   The number of experiences is the product of all enabled variations.

2. In the Percentage of visitors exposed to test section, set the percentage of visitors exposed to the test. For example:
    - Set the percentage at 40%. This means that 40% of visitors will see one of the experiences that you have created (and the original version is part of these).
    - The remaining visitors – 60% in this example – will see the original version.

    percentage

    Sitecore calculates test results based on the visitors that you expose to the test.

3. In the Statistics section, set the statistical confidence level required for the test to declare a winner. You can select 90%, 95%, or 99%. The default is 95%.
4. On the Objective tab, in the Test objective section, in the Test objective field, select what the test measures. You can select:
    - Trailing Value/Visit – the engagement value based on page views occurring after the visitor has encountered the page being tested, divided by the number of site visits.
    - Any one of the goals that have been set up in the Marketing Control Panel.

    test-objective

5. In the Select how to pick a winner field, specify the method for selecting a winner of the test:
    - Automatically select a winner based on the test objective – this is the default if the test objective is Trailing Value/Visit.
    - Automatically select a winner based on the test objective, unless it significantly decreases engagement value – this is the default for objectives that are goals. You cannot select this option for the Trailing Value/Visit objective.

      When you select this option, no winner is declared if the best experience (based on the goal) has decreased the trailing value/visit by more than 20%. Instead, the test continues.

    - Manually select winner – the user who created a test can select an experience in the Test Results dialog box and declare the experience as the winner of the test. When you choose a winner, consider the effect that the test had on the overall engagement value generated on the website.
6. In the Duration of the test section, specify the minimum and maximum time for the test to run:
    - Minimum – select 3, 7, or 14 days.
    - Maximum – select 14, 30, or 90 days.

Start a test

Note

After you start a test, you cannot modify it. For example, you cannot add or edit the test variations or the components that you are using in the test.

To start a test:

1. In the Preview and start test dialog box, click Start test.
2. To make the test go live and publish the test to your website, you must publish the page. In the Publish Item dialog, select Smart publish. We recommend that you also select the Publish related items option.

    Note

    If you use a workflow that includes Auto Publish, you do not need to publish manually.

Send feedback about the documentation to docsite@sitecore.net.

Preview variants of a component

A/B and multivariate component tests let you test alternative combinations of your website's content to find out which components or combinations of components are the most effective with visitors.

Before you run a test on the website, you can preview each of your component variations in the Experience Editor.

To preview variants of one component:

- In the Experience Editor, select the component you are testing. In the floating toolbar use the up and down arrows to preview each test variation.

    Picture 2

You can preview a multivariate test in the Preview and Start a Test dialog.

Send feedback about the documentation to docsite@sitecore.net.

Test basic content changes

When you update content on a page, Sitecore suggests that you create a content test that tests the new and old versions of your website's content to find out which content is most effective with visitors.

Note

Before you can use a test to measure the effectiveness of the content, you must set up goals with engagement value points, and assign the goals to pages or actions on your website.

When you want to compare the new version of an item with the previous version, you can start a test in the Experience Editor. This topic outlines how to:

- Set up a test
- Start a test

Set up a test

To set up a test in the Experience Editor:

1. Save and submit the item that you want to test.
2. In the notifications at the top of the page in the Experience Editor:
   - If you are using a workflow, click Approve with Test:

     message-bar

   - If you are not using a workflow and you have enabled test notifications, click Create a test:

     Picture 3

3. Enter a comment and click OK.

   The Preview and start test dialog box opens. On the Preview tab, you can preview the test and predict the outcome:

   Picture 1

Adjust test parameters

You can use the default settings for all of the test parameters. To learn about custom test configuration, you can read about contest test configuration.

Start a test

Note

After you start a test, you must not modify it. For example, do not edit the content that you are testing.

To start a test:

1. In the Preview and start test dialog box, click Start test.
2. To make the test go live and publish the test to your website, you must publish the page. In the Publish Item dialog, select Smart publish and the Publish related items option.

   Note

   If you use a workflow that includes Auto Publish, you do not need to publish manually.

Send feedback about the documentation to docsite@sitecore.net.

Path Analyzer

The Path Analyzer is an application that enables you to create a map that shows the sequential paths that contacts take as they navigate through your website. You can see the paths that contacts take while interacting with campaigns, and on their way to converting on business critical activities such as goals and events.

You can create new Path Analyzer maps to track the different types of experiences contacts have on the way to content such as campaigns or digital assets. You can also configure the Path Analyzer to track different kinds of visits to your website, such as the experience of a first time visitor or a returning customer. This information can provide you with insight into how different kinds of contacts interact with your content.

The Path Analyzer shows you which paths have the most traffic, which paths provide the highest engagement value, and which provide the highest engagement value per visit or efficiency. This enables you to gain insight into which contact experiences on your website work efficiently, and which content is less efficient and could be optimized.

Picture 3

Page Analyzer

You can use the Path Analyzer to view how contacts interact with your content on a large scale, viewing the most frequently travelled paths through your website. To view interactions on a smaller scale, you can use the Page Analyzer. The Page Analyzer shows you the paths to and from specific pages on your website.

cms_PA_pageanalysis

The Page Analyzer allows you to explore paths that lead to a page and then view the individual contacts that had this particular path experience.

cms_PA_singlepageanalysis

In this way, with Path Analyzer, you can go from the very high level path map view showing all paths, to a page-specific analysis view, down to an individual contact who had this path experience, and then pull up his or her Experience Profile:

cms_xFile_sample

Analyzing different contact experiences

The Path Analyzer lets you map and view the experiences that your contacts have on your website. You can see which pages that contacts visit on their way to triggering specific goals or page events. You can also configure paths that identify traffic coming from particular channels or particular campaigns. The Path Analyzer then shows you which paths efficiently lead contacts to desired business outcomes, and which paths are less effective.

You also have the ability to map different types of interactions that contacts have with your website. You can, for example, map the experiences of first-time visitors to your website, and then create maps that show the experiences of returning visitors and compare how the two experiences differ. You can also use the Rule Set Engine to create maps that are based on location, devices, and so on. Comparing contact experiences can give you insight into how your content works with different audiences.

Path Analyzer glossary

The Path Analyzer is an application that allows you to view the various paths that contacts take as they navigate through your websites. You can see the paths that contacts take when they convert goals and interact with campaigns, giving you an insight into which paths provide the best engagement value per conversion, and which paths are less efficient and could use optimization.

The Path Analyzer introduces several new concepts and terms.

Entry path

An entry path is the path from the entry page of a website to any other pages during a visit.

Exit node

The exit node is the end of both a visit and a path. A website visit ends when a contact closes the browser, navigates to another website, or times out.

Exit potential

The exit potential is a measurement that identifies if a path is worth continuing down to gain engagement value or if there is no additional engagement value that can be gained through optimization. If it is worth optimizing the path, exit potential indicates how much value a path could gain through optimization.

Full path

A full path shows the contact's entire visit from the entry page to the exit page.

Node

A tree consists of nodes, which are the sequence of pages in a path. The following details for each node are available in the Path Analyzer:

Node detail	Description
Subtree value	The sum of the engagement value points accrued by contacts who have visited this path.
Subtree visits	The number of visits to this path.
Subtree value per visit	The average engagement value points per visit to this path.
Exit value	The sum of the engagement value points that contacts have accrued before exiting at this node.
Exit visits	The number of contacts that left the website at this node.

Exit value per visit	The average engagement value points per visit of contacts who exited the website at this node.
Exit potential gain	The projected engagement value a path could gain through optimization.
Average time spent (sec)	The average amount of time contacts have spent on a node.
Depth	The number that describes where this node appears in its subtree or path.

Page Analyzer

The Page Analyzer is a part of the Path Analyzer that shows you the entry and exit paths that surround a specific node. You can use the Page Analyzer to examine a selected path as well as view the contacts who had this path experience.

Path

The path is a sequence of nodes that correspond to the distinct paths that contacts take during a visit.

Path root

The path root is the node that is the first page in a path.

Path value

Path value shows you the average engagement value per visit on the path.

Reports

There are four main analytics reports available in the Path Analyzer:

- *Entry Path* report - indicates the most popular entry paths through the website
- *Most Valuable Full Path* report - indicates which exit nodes provide the highest engagement value levels
- *Most Efficient Full Path* report - indicates which are the most efficient paths in terms of traffic versus engagement value
- *Optimization Candidate* report - indicates which popular paths could gain the most engagement value

Tree

The tree is the root structure of a Path Analyzer map. All paths and nodes are a part of the Path Analyzer tree.

Tree root

The tree root is the first node in a tree that initializes analytics on a website. It is the start of the Path Analyzer tree but does not represent a page on the website. A tree root can have several path roots below it.

An example of a tree root is the Internet, as contacts often enter a website from somewhere else on the Internet.

Visit

A visit is the series of actions performed by a contact from when they enter your website until they leave the website. A single contact may make multiple visits at different times. These are regarded as separate visits.

Visit value

Visit value is a metric that measures the value of a contact's actions on the website through engagement value points. You assign engagement value to actions such as downloads or purchases in the Marketing Control Panel. Visit value measures how efficiently a node generates value.

Visit value in the Path Analyzer provides you with insight into how contacts behave on your website, and the paths they take to achieve business critical goals. The more valuable the path, the more effective it is at getting contacts to convert on goals. Optimizing paths that have less visit value than anticipated can provide your company with increased contact engagement and generate value for your business.

Send feedback about the documentation to docsite@sitecore.net.

Create and configure a new Path Analyzer map

The Path Analyzer is an application that allows you to view the various paths that contacts take as they navigate through your websites. You can see the paths that contacts take when they convert goals and interact with campaigns, giving you an insight into which paths provide the best engagement value per conversion, and which paths are less efficient and could use optimization.

You can create new Path Analyzer maps to track the paths that contacts have followed from specific channels or campaigns, or to particular goals or events. You can also configure the Path Analyzer to track different kinds of visits to your website. This information can help you see precisely how contacts navigate through your website.

You can also gain insight into which paths to certain experiences provide the highest engagement value level, and which paths could be optimized. Optimization recommendations can help you refine your marketing strategy and how you present content to your contacts, enabling you to convey your content in a more meaningful way.

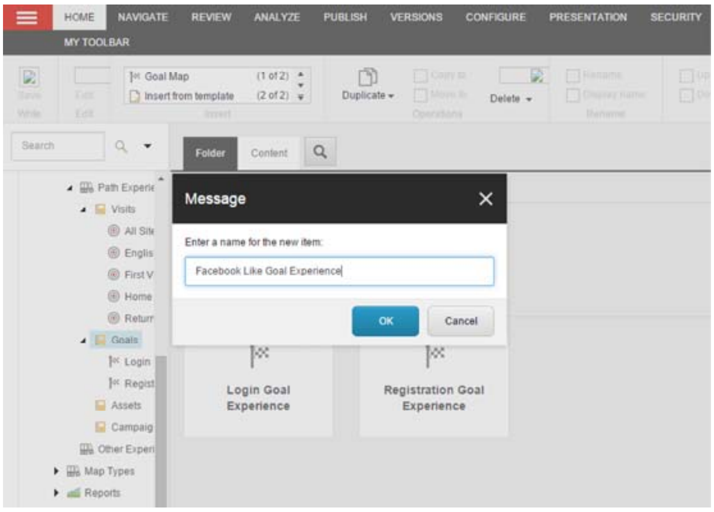
You create and configure new Path Analyzer maps in the Marketing Control Panel.

1. In the content tree, in the Marketing Control Panel, expand the *Path Analyzer* node.

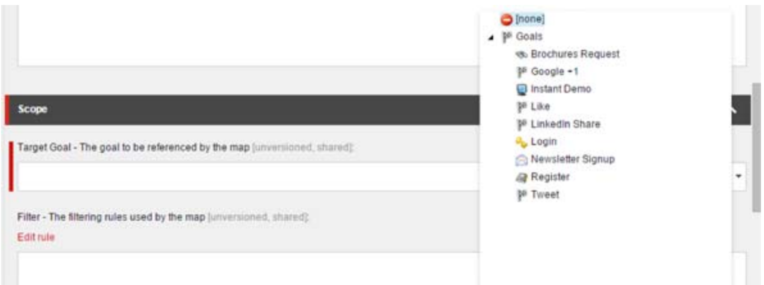
2. Expand the *Maps* node, then click *Path Experience Maps*. Expand the *Path Experience Maps* node, then select one of the four categories to create a new Path Analyzer map.

Category	Comments
Visits	A visit map that shows the different experiences that contacts can have on your site, which are not based on an asset, campaign, or goal, for example, a first time visit.
Goals	A map configured around converting a specific goal. This is convenient for users who want to see maps of interactions leading to the selected goal.
Asset	A map configured to show the paths that contacts take to download or interact with a particular digital asset, such as a PDF or video.
Campaign	A map that shows path experiences for contacts coming from a specific campaign.

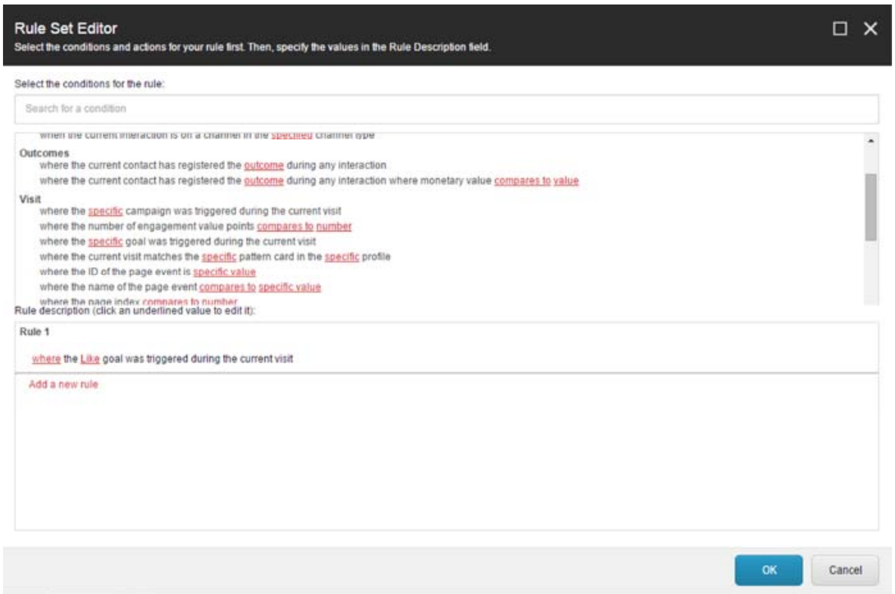
3. After you have selected a category, on the Home tab, in the Insert group, add a new path map, for example, *By Campaign*. In the dialog box, give the new map a name, then click OK.



4. On the Content tab, under Scope, click on the Target Goal/Media Asset/Campaign field. This opens up a content tree directory. Click the item you want to associate with this map.



5. On the Content tab, under Scope, you can enter start and end dates for the map.
   - To enter the start date for the map, under Start, click *Today*. The current day is the default selection. This means that the map will add all visits after the current day. To select a different day, click the drop-down arrow to open a calendar dialog box and select a date from the calendar.
   - To enter the end date for a map, under End, click *Today*. To select a different day, click the drop-down arrow to open a calendar dialog box and select a date from the calendar.
6. On the Content tab, under Scope, click *Edit Rule* to open the Rule Set Editor. In the Rule Set Editor dialog box, select the rule you want to associate with this experience map. Once you have configured the actions and conditions for the rule, click OK. In this example, the Facebook Like goal is being associated with the new map.



7. To deploy the new map, on the Review tab, in the Workflow group, click Deploy and then click Save.

   Note

   You deploy new maps using a workflow process. Once you have deployed a new map, it enters a system queue within the xDB. The system processes new maps once every 24 hours, so if you have multiple new maps, you should deploy them simultaneously using the Deploy action within the Workbox.

   Please consult your Sitecore System Administrator if you do not see the data for your new maps after 24 hours.

Send feedback about the documentation to docsite@sitecore.net.

Path Analyzer maps

The Path Analyzer is an application that allows you to view the various paths that contacts take as they navigate through your websites. You can see the paths that contacts take when they convert goals and interact with campaigns, giving you an insight into which paths provide the best engagement value per conversion, and which paths are less efficient and could use optimization.

Maps are a visual representation of the paths that contacts take through your website. Maps consist of three basic elements:

- Tree
- Path
- Nodes

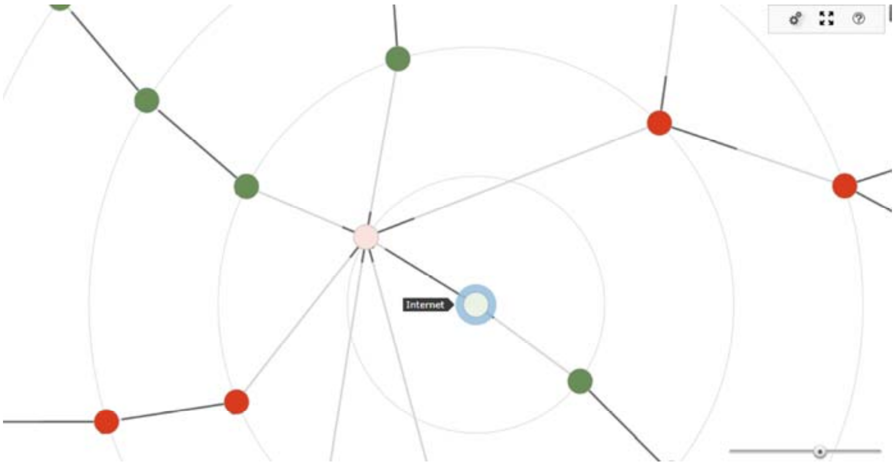
The tree is the basic structure of a Path Analyzer map. A tree consists of many paths, or branches, radiating out from the map's central node. This central node is also known as the tree node.

Paths consist of groups of nodes that represent the sequential pages that contacts have interacted with on their way from specific channels or campaigns, or to particular goals or events. Two or more contacts must have had the same sequence of interactions for a path to be created.

Understanding paths

Paths provide you with information about how efficiently your website works. This information can give you insight into which content generates the highest engagement value, and which content is not performing as efficiently as it could.

Path length refers to the length of the connecting line between nodes. Path length signifies the amount of traffic that the node receives. The longer the shaded portion of the path between nodes is, the more traffic that particular node receives. If, for example, you have a node that receives a lot of traffic but does not generate much engagement value, it would be a candidate for closer examination and optimization.



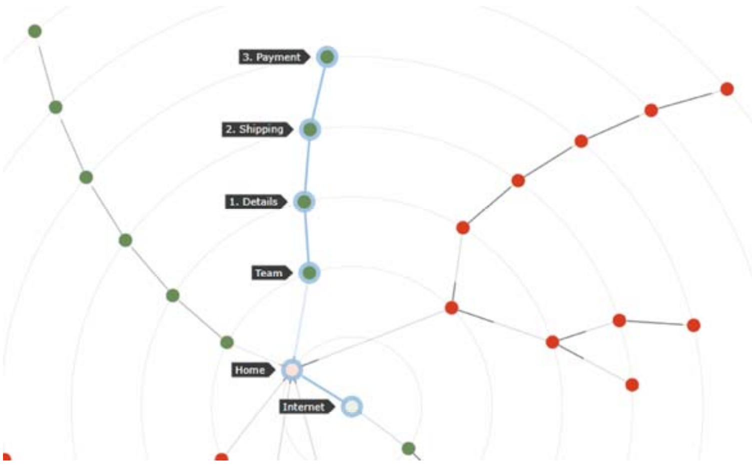
Paths and efficiency

You can measure the efficiency of your website using Path Analyzer maps. On a map, the nodes along the map show you the relative engagement value for that particular page. Nodes range in color from dark red to dark green. The greener the nodes along a path are, the more efficient the path is at generating engagement value. A path containing red nodes indicates that the pages generate low engagement value and could be candidates for optimization

Another key to understanding path efficiency is node size. Node size increases as the engagement value generated by contacts who pass through the node increases.

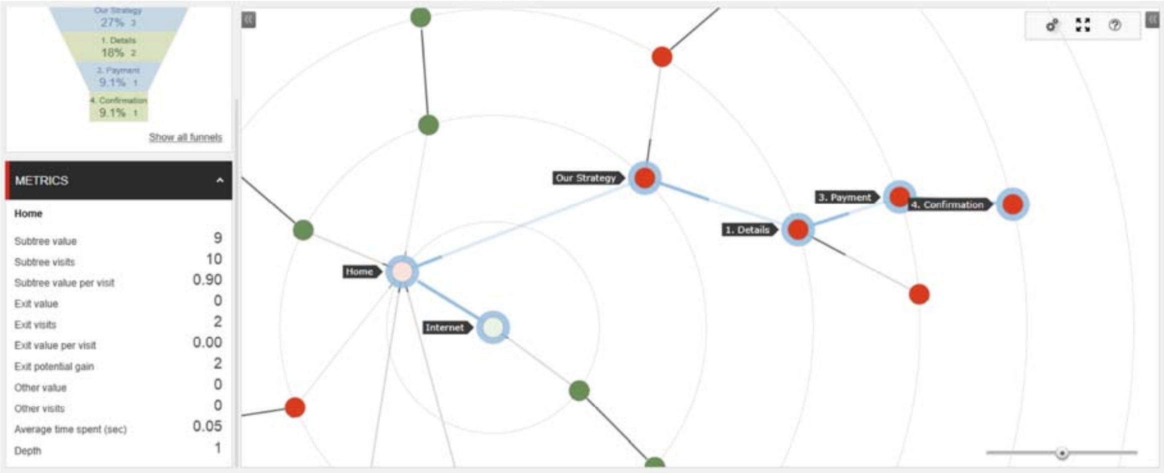
For example, you could view a Path Analyzer map that has a high traffic path with dark red nodes. This would indicate that the path received a lot of contact traffic, but had low engagement value levels. You could see that other paths that contained certain types of pages, such as employee biographies, lead to higher conversion rates than the more frequently used paths.

By examining what works on your website and then steering visitors towards the most efficient paths, you have a greater chance of generating more engagement value. This helps you make your website and your marketing strategies more efficient.



Analyzing maps

To the left of every Path Analyzer map, you can view analytics details. You can filter the analytics to view different types of Path Analyzer reports using the Metrics panel on the Path Analyzer map.



The Metrics panel contains the following categories:

Item	Description
Subtree value	The sum of the engagement value points accumulated by contacts who have visited this path.
Subtree visits	The number of visits to this path.
Subtree value per visit	The average engagement value points per visit to this path.
Exit value	The sum of the engagement value points that contacts have accrued before exiting at this node.
Exit visits	The number of contacts that left the website at this node.
Exit value per visit	The average engagement value points per visit of contacts who exited the website at this node.
Exit potential gain	The engagement value a path could potentially gain through optimization.
Average time spent (sec)	The average amount of time contacts have spent on a node.
Depth	The number that describes where this node appears in its subtree or path.

If you have not selected a node within a path, the Metrics panel will show information about the Internet root node. To select a new node, hover or click on it. This will show analytics information, such as subtree visits and value, for both the node and the path leading up to the node.

Send feedback about the documentation to docsite@sitecore.net.

View a path funnel

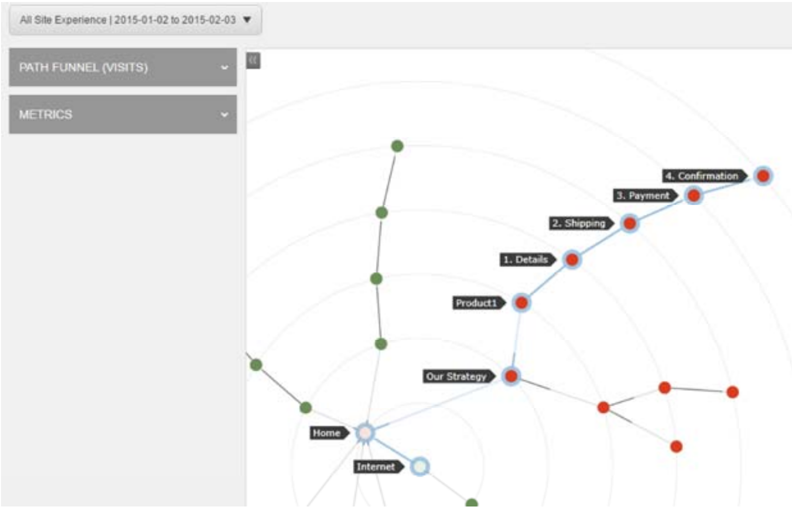
The Path Analyzer is an application that allows you to view the various paths that contacts take as they navigate through your websites. You can see the paths that contacts take when they convert goals and interact with campaigns, giving you an insight into which paths provide the best engagement value per conversion, and which paths are less efficient and could use optimization.

The path funnel shows you how many contacts leave a path and at which specific nodes they exit. From the path funnel, you can access the *Funnel* report. This shows you the value, visits, and value per visits of all contact experiences through this path.

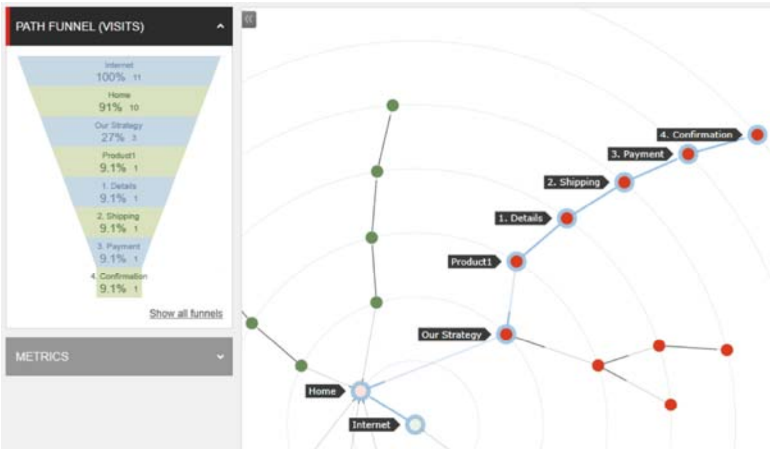
The path funnel and the *Funnel* report provide you with valuable insight into where contacts leave your website, which can help you analyze where your website loses potential conversions and where you can optimize your content strategy.

To view a path funnel:

1. From the Sitecore Launchpad, open the Path Analyzer.
2. Choose a site map from the drop-down menu
3. On the map, click the end node of the path for which you want to view a funnel.



4. Under Path Funnel (Visits), expand the report to view the funnel for this path.



5. Click Show all funnels to view the funnel*Funnels* report. The *Funnels* report shows you the value, visits, and value per visits of all experiences through this path. You can also view where contacts dropped out of this experience funnel.

   To save a local version of this path funnel, click Export image.

Send feedback about the documentation to docsite@sitecore.net.

Items

Items are the basic building blocks of your Sitecore website. An item can represent any kind of information that makes up a webpage, for example, a piece of text, a media file, a layout etc.

An item always has a name, an ID that uniquely identifies the item, and it is based on a template that defines which fields the item contains. In addition, an item can have multiple versions and multiple language versions.

You can use the Content Editor or the Experience Editor to edit, create, and insert items.

- In the [Content Editor](#), all the items on your website are stored in the content tree, where you can navigate to a specific item in a hierarchy.

  In the content tree, the relationships between an item and its associated items are as follows:

- ◦ Parent – the item directly above the item
- ◦ Sibling – an item at the same level as the item
- ◦ Subitem – an item directly below the item.
- ◦ Ancestors – all the items that are above the item
- ◦ Descendants – all the items that descend from the item

cms_ce_ContentTree_ItemRelationships

- In the Experience Editor, the items are displayed as they look on your website. The Experience Editor is a WYSIWYG (What You See Is What You Get) editor that allows you to edit items directly on the page.

  Note

  The ways in which you can create and edit items in Sitecore can vary depending on your security settings and the way your website is setup.

In both the Content Editor and the Experience Editor, an item must be in the editing state before you can edit it. You can lock an item, before you start to work on it to ensure that no other content authors work on the item at the same time.

Send feedback about the documentation to docsite@sitecore.net.

Copy or duplicate an item

In the Content Editor, you can create a new item by copying or duplicating an existing item:

- Copy an item – copies the selected item and all its content and subitems to a location in the content tree that you specify.
- Duplicate an item – copies the selected item and all its content and subitems to the same location in the content tree.

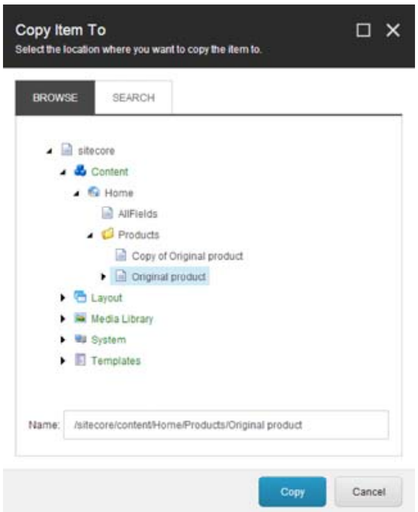
Note

In Internet Explorer, you can also copy, cut, and paste items to and from the clipboard. The clipboard functionalities are available on the Home tab, in the Clipboard group, but due to JavaScript restrictions in other browsers, the Clipboard group appears only in Internet Explorer.

Copy an item

To copy an item to another location:

1. In the content tree, locate and select the item that you want to copy.
2. On the Home tab, in the Operations group, click Copy to. Alternatively, you can also right-click the item in the content tree, click Copying and then Copy to.
3. In the Copy Item To dialog box, select the location where you want the copy the item to be stored.



   Note

   You can navigate to the location through the content tree on the Browse tab, or you can search for the location on the Search tab.

4. Click Copy to create a copy of the item in the specified location.

Now you can edit the new item, add subitems, or move the item to another location.

Note

To copy multiple items at the same time, you must run a search for all the items that you want to copy and use the *Copy results to* search operation.

Duplicate an item

To duplicate an item:

1. In the content tree, locate and select the item that you want to duplicate.
2. On the Home tab, in the Operations group, click Duplicate. Alternatively, you can right-click the item and click Duplicate.
3. Enter a name for the new item and click OK.

Now you can edit the new item, add subitems, or move the item to another location.

Send feedback about the documentation to docsite@sitecore.net.

Create an item in the Content Editor

When you create a new item in the Content Editor, you must first select an item in the content tree to indicate where you want the new item to be located. You can create a new item as a sibling or a subitem to the item that you select:

- A sibling is an item that you create at the same level as the selected item.
- A subitem is an item that you create under the selected item.

This topic outlines how to

- Insert a new item
- Insert an item based on a template

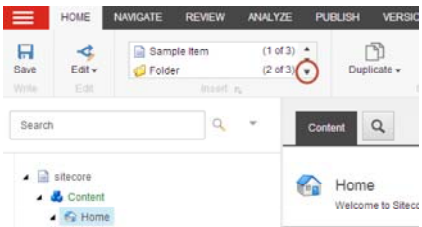
Tip

You can also create new items by copying or duplicating existing items.

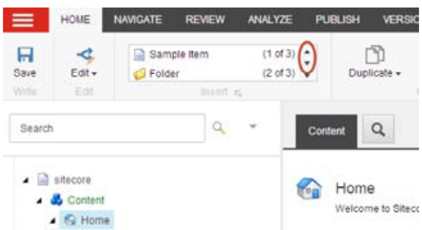
Insert an item

To insert a new item:

1. In the Content Editor, navigate to the item where you want your new item to be located.
   - To insert a sibling to the selected item, on the Home tab, in the Insert group, click the drop-down arrow and in the Insert a new sibling section, select the item type that you want to insert.



   - To insert a subitem to the selected item, on the Home tab, in the Insert group, use the up and down arrows to look for and select the type you want to use, enter the name of the new item, and click OK.



   Alternatively, you can right-click the item in the content tree, click Insert, and then click one of the suggested item types.

2. Enter a name for the new item and click OK.

   Every new item that you create automatically locks and opens in the Content Editor. You must unlock and save your items before other users can work on them.

3. As appropriate, edit the new item, add subitems, or move your item to another location.

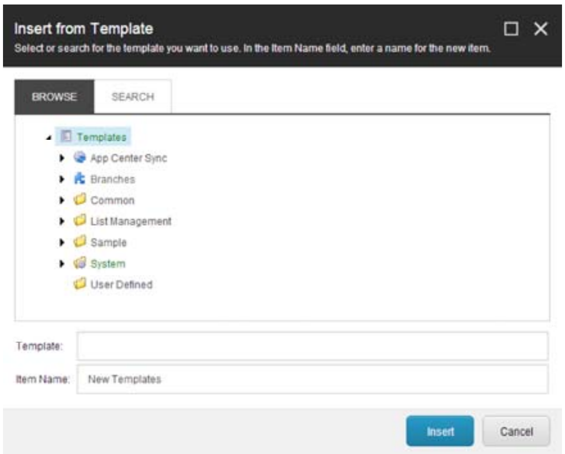
Insert an item based on a template

If you have sufficient access rights, you can create a new subitem from any template that is available in your Sitecore installation.

To insert an item from a template:

1. In the content tree, find and select the item that you want to create a subitem for.
2. On the Home tab, in the Insert group, click the drop-down arrow and click Insert from template.

3. In the Insert from Template dialog box, navigate to the template that you want to base the new item on.



Alternatively, you can right-click the item in the content tree, click Insert, and then click Insert from template.

Note

To locate the template you can navigate through the content tree on the Browse tab, or you can use the Search tab to search for the item.

4. In the Item name field, enter a name for the new item and click Insert.
5. As appropriate, edit the new item, add subitems, or move it to another location.

Important

You can create an item in an item bucket in exactly the same way as you create a regular item. If the template that the new item is based on is bucketable or if you make the item bucketable, the item is hidden in the item bucket. If the item or the item's template is not bucketable, the item is visible in the item bucket. Consequently, you can have an item bucket that is a hybrid of hidden items and visible items..

Send feedback about the documentation to docsite@sitecore.net.

Create an item in the Experience Editor

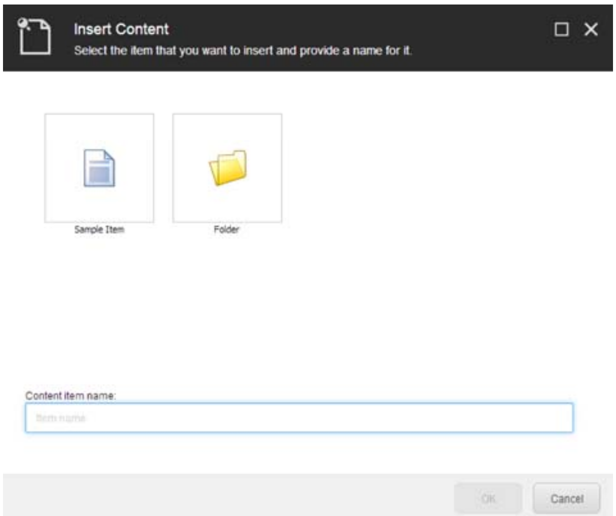
You can create new items and insert them directly on the webpage using the Experience Editor.

Note

How you create items in Sitecore can vary depending on the security roles that you have and the way your website is set up.

To insert a new item:

1. In the Experience Editor, navigate to the page that you want to add a new item to.
2. In the ribbon, on the Home tab, in the Edit group, click Insert page.
3. In the Insert Content dialog box, select the type of item that you want to insert, and in the Content item name field, enter a name for the item.



The type of item that you can create varies depending on where you are on the website. For example, if you are on the front page, you can insert a new section and you can add appropriate items to that section.

4. Click OK. The new section containing no items appears on the website and you can now edit the item by adding text, images, sidebar text, and so on.

Note

When you create a new item, it is automatically locked to prevent other users from editing it.

5. When you are finished editing the new item, click Save and then Unlock to make the item available to other content authors.

Send feedback about the documentation to docsite@sitecore.net.

Delete an item

Inevitably, you will need to delete content items from time to time. In the Content Editor or the Experience Editor, you can delete an item and all of its versions and subitems, or you can delete a single version of an item.

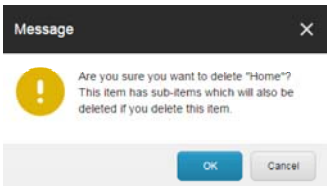
This topic outlines how to:

- Delete an item and its subitems and versions
- Delete a version of an item
- Delete multiple items

Delete an item, its subitems and versions

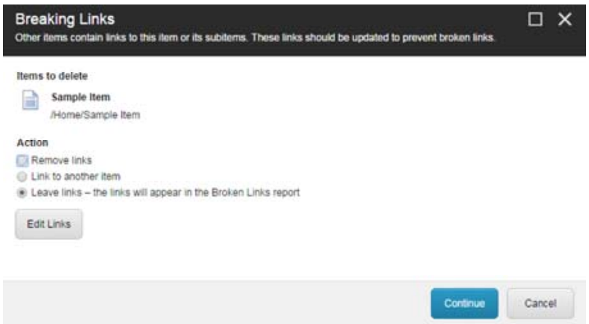
To delete an item and all its subitems and versions:

1. In the Content Editor or the Experience Editor, navigate to the item or page that you want to delete.
   - In the Content Editor, on the Home tab, in the Operations group, click Delete. Alternatively, you can right-click the item in the content tree, and click Delete.
   - In the Experience Editor, on the Home tab, in the Edit group, click Delete page.
2. If the item contains subitems, a message appears informing you that that the item contains subitems and that these items will be deleted too.
   - To save any of the subitems, click Cancel and move them to another location.
   - To delete the item and its subitems, click OK. This deletes every version of the item in every language.



If there are links to the item you delete, you must decide how to deal with the broken links in the Breaking Links dialog box that opens.

1. In the Breaking Links dialog, you can:
   - Remove links – remove all the links that exist to this page.
   - Link to another item – change all the links to this page to link to another item.
   - Leave links – manage the broken links in the Broken links report.



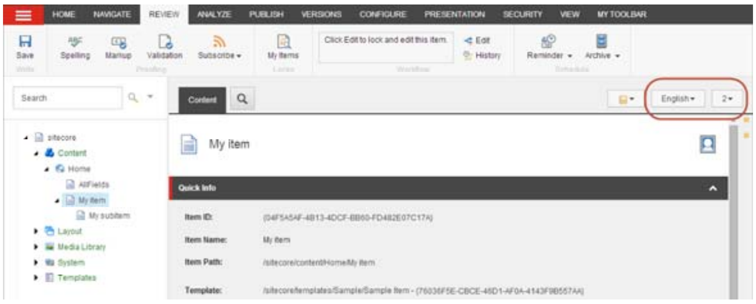
2. To open the item that links to the item you want to delete and edit the link directly from there, click Edit Links. In the Edit Link dialog box, click Edit.
3. In the Breaking Links dialog box, click Continue.
4. If you selected Link to another item, in the content tree, click the item that you want to link to instead. To update the link, click OK.

Delete a version of an item

If you have created several different versions of an item, you can delete individual versions of the item without deleting the item or any of its other versions or subitems.

To delete a version of an item:

1. In the content tree, locate the item that has a version that you want to delete.
2. If the item is available in multiple languages or versions:
   - Click the Language button and select the language version that you want to delete.



- Click the Versions button and select the version that you want to delete.

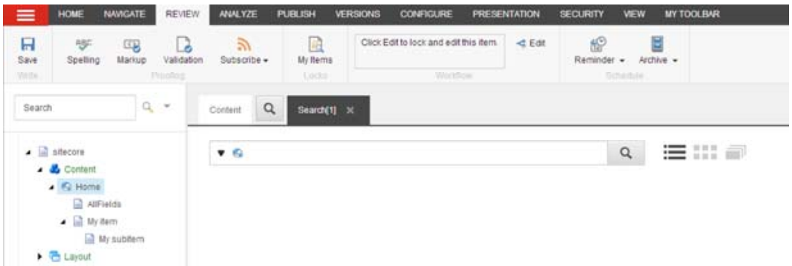
3. On the Versions tab, in the Versions group, click Remove. The version that you have selected is deleted.

Delete multiple items

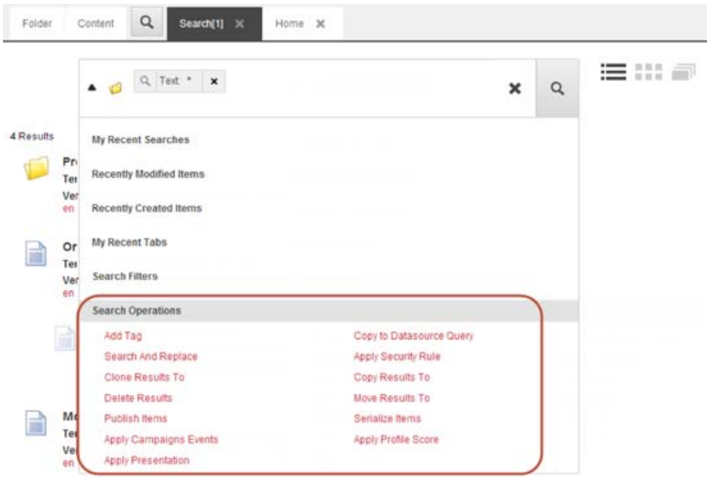
In the Content Editor, you can use the [Sitecore search](#) functionality to delete multiple items at the same time.

To search for and delete multiple items:

1. In the Content Editor, locate and select an item that you want to delete.
2. Click the Search tab, and in the Search field, enter a search keyword for the items that you want to delete.



3. Use the necessary search facets or filters to narrow down your search so that the search result only displays the items that you want to delete.
4. Click the drop-down arrow ▼ to the left of the search field, click Search Operations, and then click Delete Results.



Warning

Once you click Delete results, the result items are deleted without a warning or confirmation. However, if you regret deleting an item, you can open the [Sitecore Recycle Bin](#) where you can restore any deleted item.

Send feedback about the documentation to docsite@sitecore.net.

Edit content directly on the page

The Experience Editor is a WYSIWYG editor that allows you to easily make changes to items directly on the page. You can edit all the items that are visible on the page — text, graphics, logos, links, and so on.

The Experience Editor, you can edit the following field types directly on the page:

- Simple text fields – this field can only contain simple text, such as single-line and multi-line text, dates, and links.
- Rich Text fields – this field gives you more control over how the text appears on the webpage. You can edit a Rich Text field directly on the page or open the Rich Text Editor to get access to more formatting functionality.
- Word fields – this field allows you to use Microsoft Word to edit the text. Depending on the configuration of your website, you can edit the Word field in Microsoft Word or directly on the page. If your website supports inline editing, all of the features of Microsoft Word are also available in the inline editing pane.

- Image fields – a field that only contains images. In an image field, you can change the image, modify the image appearance, or remove the image directly on the page.

This topic outlines how to:

- Customize the Experience Editor view
- Edit a field directly on the page

Customize the Experience Editor view

If you have the appropriate access rights, you can change the way the Experience Editor presents the editable items for you.

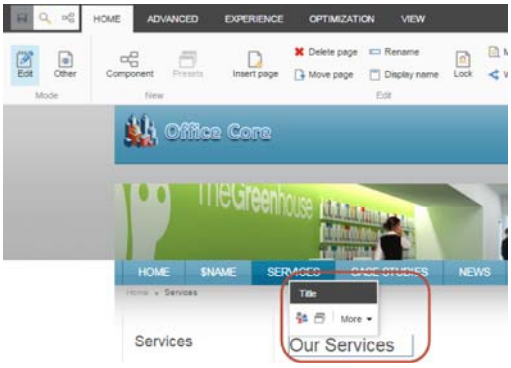
On the View tab, you can setup the Experience Editor:

- To show the editable text fields, select the Editing check box. When you move your mouse over the fields in the Experience Editor, the text fields appear with a dotted line.
- To show the renderings and placeholders, select the Designing check box. When you move your mouse over the fields in the Experience Editor, the renderings and placeholders appear with a dotted line.
- To show outlines of all the objects on a page, select the Controls check box. Use this feature to understand how the page is organized. This check box is only active when the Designing check box is selected.

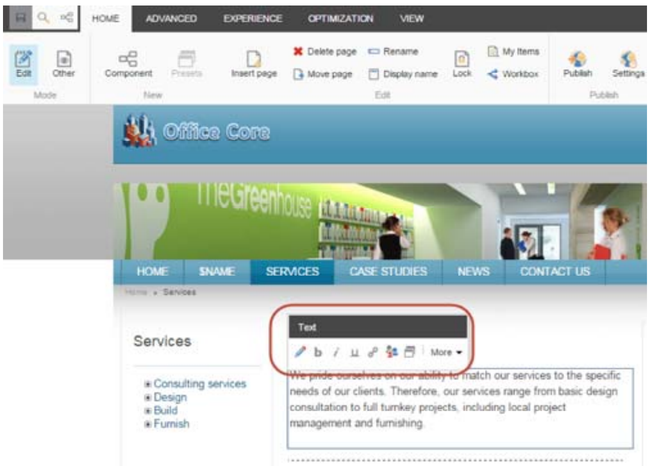
Edit a field directly on the page

To edit a field directly on the page:

1. Log in to the Experience Editor and navigate to the item that you want to edit.
2. Click the item and enter the appropriate text. Depending on which type of field you select, you have different options in the floating toolbar that appears:
   - In a simple text field, you can only enter text. The field automatically expands to fit the text you enter.



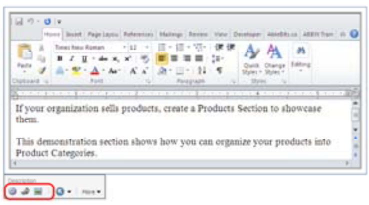
   - In a rich text field, you can perform simple formatting on the text and insert an image or a link.



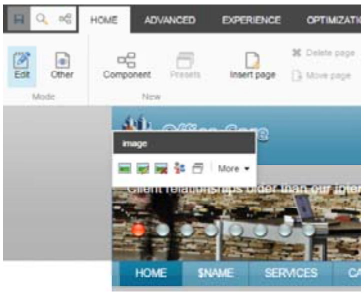
Note

The options available in the floating toolbar vary depending on your version of Sitecore CMS and your access rights.

   - In a Word field, you can insert an image or a Sitecore link, and choose how you want the Word ribbon displayed:



   - In an image field, you can insert various types of media. These could be animations, film clips, sound files, and so on.



3. If you want to edit the item's field values in the Content Editor, click the More drop-down arrow and click Edit the related item. This opens the item in the Content Editor above the Experience Editor in the same browser.

   To return to the Experience Editor, click the Sitecore Start button and select Exit.

4. When you have finished editing the field, click Save to save your changes.

Send feedback about the documentation to docsite@sitecore.net.

Edit the layout of an item

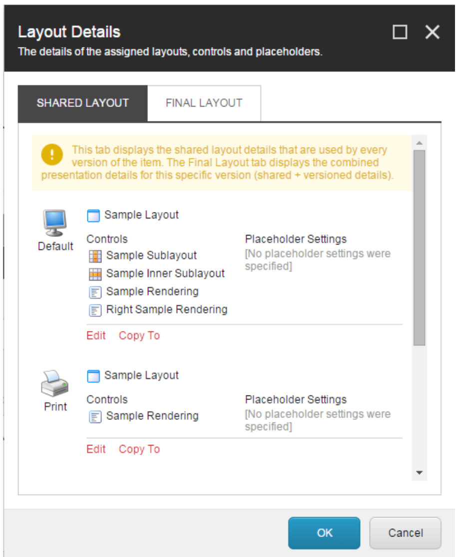
Versioned layouts enables you to:

- Specify the layout of an item
- Reset the layout of an item

Specify the layout of an item

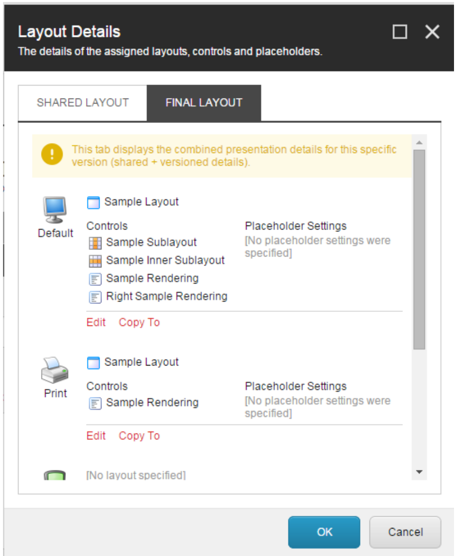
In the Content Editor, you can specify shared and versioned layouts of an item:

1. On the Presentation tab, click Details.
2. In the Layout Details dialog, on the Shared Layout tab, edit the layout details that apply to all the versions of the item.



3. Click OK to save and close the dialog.
4. In the Content Editor, select a language and version of the item.

5. Open the Layout Details again and on the Final Layout tab, edit the layout details for the language version of the item.



Sitecore only uses the layout you specify on this tab for the selected version and language.

Reset the layout of an item

You can reset the layout of an item. When you reset a versioned layout, the final layout becomes the same as the shared layout of the item.

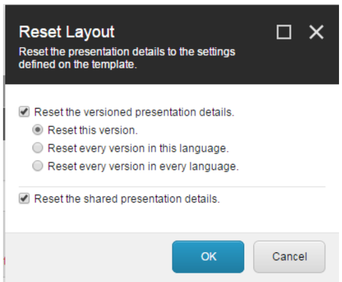
You can reset versioned layouts in these ways:

- Reset only the versioned layout for the current version of the item.
- Reset the versioned layout for all versions in the currently selected language of the item.
- Reset the versioned layout for all versions in all languages.

You can reset a shared layout to the one defined in the standard values item of the data template that the item is based on.

To reset the layout of an item, in the Content Editor:

1. On the Presentation tab, click Reset.



2. In the Reset Layout dialog, select either Reset the versioned presentation details or Reset the shared presentation details, or both, and click OK.

Send feedback about the documentation to docsite@sitecore.net.

Edit the website content

When you edit content in rich text fields, you can benefit from the advanced editing functionalities in the Rich Text Editor or in Microsoft Word. For minimal editing functionality, you can edit rich text fields directly on the item or page in the Experience Editor.

The Rich Text Editor

The Rich Text Editor is the standard text editor in Sitecore. In this editor, you can edit all the content in a rich text field.

In the Rich Text Editor, you can:

- Paste text from Microsoft Word.
- Find and replace text.
- Insert pictures or other media items.
- Insert and delete links to other parts of the website and to external websites.
- Insert symbols.

Note

The Rich Text Editor is a standard word processing tool, but in Sitecore, most of the formatting functionality has been removed because the style sheets and templates for a website determine the format of the items that content authors create and edit.

Microsoft Word

You can use Microsoft Word as your text editor in Sitecore and benefit from all the functionality that is available in Microsoft Word. You can open the Word field editor from both the Content Editor and the Experience Editor.

- Insert a link to an item in Sitecore.
- Insert an image from the Media Library.

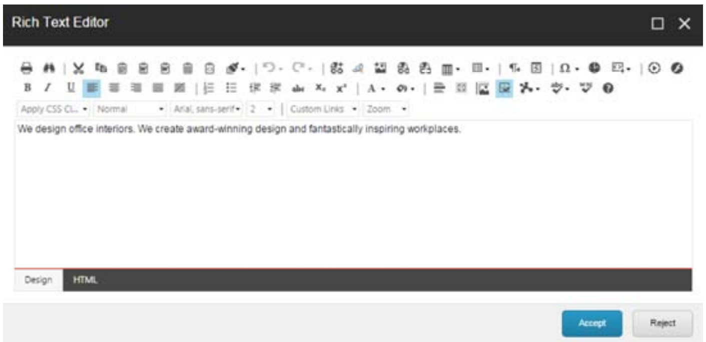
Important

To edit a text field in the Word field editor, your administrator must set up the text field as a Word field. In addition, make sure to use Microsoft Word 2007 or later and Internet Explorer 7 or later. If you use Internet Explorer 8, you must configure your security settings to support Word fields in Sitecore.

Edit a text field

To edit a text field in the Content Editor or in the Experience Editor:

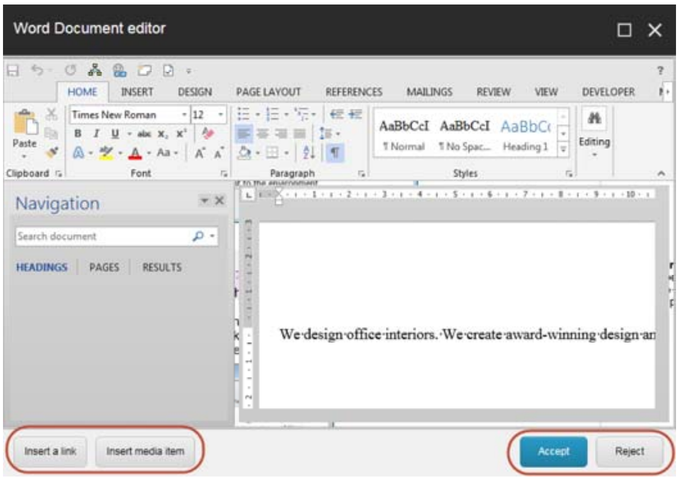
1. In the Content Editor or the Experience Editor, navigate to the item or page that you want to edit. Make sure to select the relevant version and language of the item.
2. To open the text editor:
   - From the Content Editor – in the content area, locate the Rich Text field you want to edit and click Show Editor. If the text field is a Word field, you click Edit.
   - From the Experience Editor – click the field that you want to edit and in the floating toolbar that appears, click Edit the text. If the text field is a Word field, click Edit the word document.
3. Make the appropriate changes to the content in the text field.
   - In the Rich Text Editor, if you prefer to edit the content in HTML format, click HTML.



   Note

   Your Sitecore system administrator controls the features that are available to you. If you need more features, ask your administrator to make them available.

   - In a Word field, to insert a Sitecore link to another item on your website, click Insert a link. To insert an image from the Media Library, click Insert media item. If the image is not available in the Media Library, you can use the insert picture functionality in Microsoft Word.



   Note

   Images that you want to use regularly, you should upload to the Media Library.

4. When you are finished editing the content, click Accept to save your changes.

Note

To make sure that links do not break when items are, for example, renamed, deleted, or moved, all the links to Sitecore items or media items should have the format:

```
<img ... src="-/media/{ITEM_ID}.ashx" />
```

or

```
<a href="~/link.aspx?_id={ITEM_ID}&amp;_z=z">
```

For example, `<img alt="logo" height="51" width="204" src="-/media/094AED0302E7486880CB19926661FB77.ashx" />`

Send feedback about the documentation to docsite@sitecore.net.

Lock and unlock an item

In Sitecore, you can lock the item you are working on to prevent other users from editing the same item at the same time. If the item that you want to edit is already locked, you cannot edit it, you can only view it. When you have finished editing the item, you must unlock the item to make it available to other users again.

Depending on the set up of your Sitecore installation, you may be prompted to lock the item, before you can edit it.

A Sitecore administrator can have the authority to edit another user's locked items. Therefore, an administrator and another user can edit an item at the same time. If a user saves changes to an item, and another user saves changes to the same item afterwards, a message box appears, asking the user if the previously saved changes should be overwritten.

Important

Locking an item does not keep it from being published. If you are working on an item and you want to keep it from being published, you need to set up the publishing restrictions for the item.

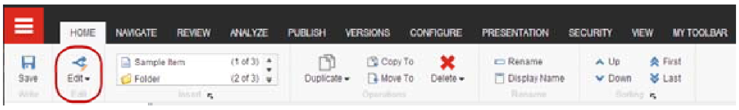
This topic outlines how to:

- Lock an item
- Unlock an item

Lock an item

To lock an item manually:

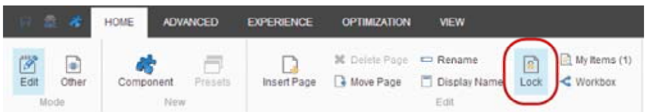
1. In the Content Editor or the Experience Editor, navigate to the item or page that you want to edit.
   - In the Content Editor, on the Home tab, click Edit to lock the item.



   Note

   In the content tree, you can use the Quick actions to see a quick overview of the items that you have locked. Right-click left of the content tree and select My Locked Items.

   - In the Experience Editor, on the Home tab, in the Edit group, click Lock.



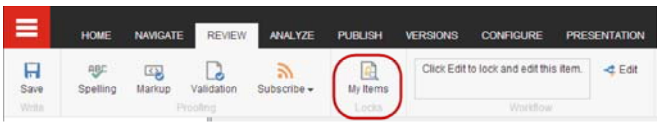
2. When you are finished editing the item, make sure to unlock the item again.

Unlock an item

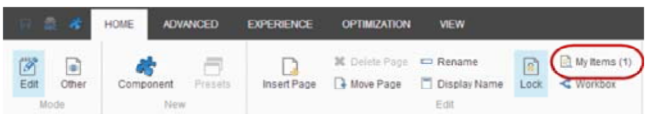
If you have locked and edited numerous items, you can lose track of them. Sitecore keeps track of all the items that are locked, and by who. In this way, you can easily lock one or more items at the same time.

To unlock an item

1. To see a list of the items that are locked by you, or to unlock one or more items:
   - In the Content Editor, on the Review tab, in the Locks group, click My Items.



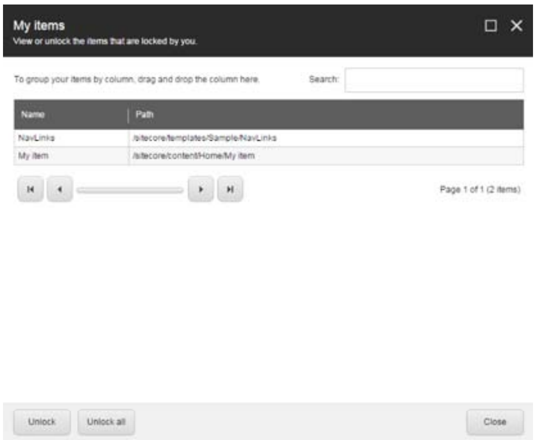
   - In the Experience Editor, on the Home tab, in the Edit group, click My Items.



   Note

   The number next to the My Items button on the ribbon indicates the number of items that you have locked.

2. In the My Items dialog box, select the items that you want to unlock and click Unlock. To unlock all items at the same time, click Unlock All.



Send feedback about the documentation to docsite@sitecore.net.

Paste content from Microsoft Word into a Rich Text field

If you prefer writing in Microsoft Word, you can write your content in Microsoft Word first and then paste it onto your website. You simply copy your text from Microsoft Word and insert the text in a field by pressing CTRL+V on your keyboard.

If you need to paste the content from Microsoft Word into a Rich Text field, you need to use the Paste Word functionality that is available in the Rich Text Editor.

Important

When you paste your Microsoft Word content into the Rich Text Editor, all the formatting is removed. For example, you lose the text font and size.

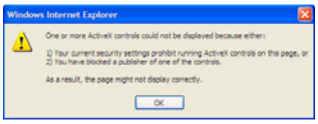
To paste content from Word into a Rich Text field:

1. In Word, select and copy all the text that you want to paste into a Rich Text field in Sitecore.
2. In Sitecore, navigate to the item that you want to edit and do one of the following:
    - From the Content Editor – in the content area, locate the Rich Text field that you want to edit and click Show editor.
    - From the Experience Editor – click the Rich Text field that you want to edit and in the floating toolbar, click Edit text.
3. Click Paste Word to insert the text into the Rich Text Editor.
4. When you are finished editing in the Rich Text Editor, click Accept to save your changes.

Send feedback about the documentation to docsite@sitecore.net.

Troubleshooting: Word fields

When you use Internet Explorer 8, the first time you access a Word field in Sitecore, you receive an error that tells you that your security settings do not allow you to run ActiveX controls on the page. This error occurs because you need to configure your security settings for Internet Explorer.



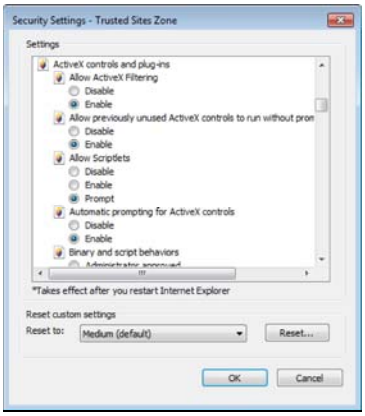
If you use Internet Explorer 8 for editing in Sitecore and want to edit your content in Microsoft Word, you must configure your Internet options to support Word fields in Sitecore.

Solution

To configure Internet Explorer 8 to support Word fields in Sitecore:

1. Ensure that you have only one instance of Internet Explorer open.
2. In Internet Explorer, click Tools (ALT+T), and then click Internet Options.
3. In the Internet Options dialog box, on the Security tab, select Trusted Sites, and in the Security level for this zone section, click Custom level.

4. In the Security Settings – Trusted Sites Zone dialog box, in the Settings section, scroll down to the ActiveX controls and plug-ins node.



5. In the Automatic prompting for ActiveX controls section, click Enable.
6. Click OK to save your changes.
7. When you access a Word field in Sitecore, you are prompted to install an ActiveX component.

8. Click Install to install the component.

When the component is installed, you can edit the field in Microsoft Word.

Note

After you have installed the ActiveX component, you should consider changing the security settings back to the original settings.

Send feedback about the documentation to docsite@sitecore.net.

Create an anchor and link to it

An anchor is an HTML code that is used as a bookmark to create a link to a particular section within a page, for example, it could be a *Back to top* link that takes the user to the top of the current page.

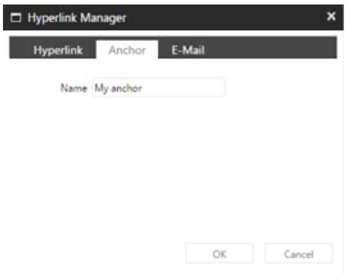
In Rich Text fields, you can use the Hyperlink Manager to create an anchor in your text, and then to create the link to your anchor.

Note

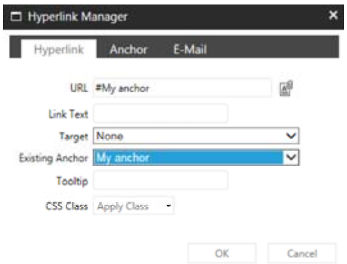
In Word fields, you cannot access the Hyperlink Manager. Instead, you must use the Hyperlink functionality within Microsoft Word.

To create an anchor:

1. In the Experience Editor or the Content Editor, navigate to the item where you want to insert an anchor.
2. Open the Rich Text Editor and select the text or image that you want make an anchor.
3. Click the Hyperlink Manager .
4. In the Hyperlink Manager dialog, click the Anchor tab and in the Name field, enter a unique name for the anchor.



5. Click OK.
6. To create a link to this anchor, you create a new link with the Hyperlink Manager. On the Hyperlink tab, in the Existing Anchor field, choose the anchor you created and click OK.



Send feedback about the documentation to docsite@sitecore.net.

Insert a link in a text field

In Rich Text fields or Word fields, you can use the Insert Sitecore Link functionality or the Hyperlink Manager to create a link on either an image or a text string.

Note

To make sure that links do not break when items are, for example, renamed, deleted, or moved, all the links to Sitecore items or media items should have the format:

```
<img ... src="-/media/{ITEM_ID}.ashx" />
```

or

```
<a href="~/link.aspx?_id={ITEM_ID}&amp;_z=z">
```

For example, `<img alt="logo" height="51" width="204" src="-/media/094AED0302E7486880CB19926661FB77.ashx" />`

This topic outlines how to insert a link with:

- The Sitecore Links functionality
- The Hyperlink Manager

The Sitecore Link functionality

If you want to insert a link to an existing item or page on your website, you must use the Insert Sitecore Link functionality. This functionality lets you search for the item you want to link to which is very useful if for example the item is a hidden item in an item bucket.

You can insert a link with the Insert Sitecore Link functionality in both Rich Text fields and Word fields. You can insert a link to:

- An item on your website – for example, to a different page on your website.
- A media item – for example, a video or a PDF file in your Media Library.

Note

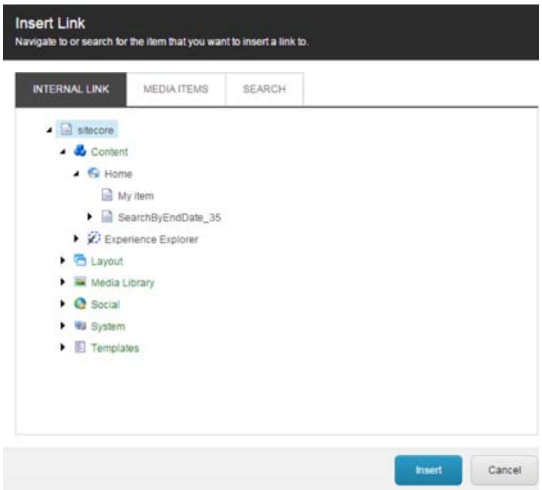
In the Experience Editor, you can only edit links on images that are in a Rich Text field or in a Word field. To edit the link you must open the text editor.

To insert a link with the Sitecore Link:

1. In the Content Editor or the Experience Editor, navigate to the text field where you want to insert a link and open the text editor:
2. In the text editor, select the text or image that you want to be a link and to insert a link in:
    - A Rich Text field, click Insert Sitecore Link.
    - A Word field, click the Insert a Link button.
3. In the Insert a Link dialog, select the item that you want to link to and click Link.

    Note

    To locate the item, you can navigate through the content tree on the Internal Link tab or on the Media items tab, or you can search for the item on the Search tab.



To remove the link, select the link in the text editor and click Remove Link. Then, the link is removed and the text turns into normal text.

If you want to change an existing link entirely, you simply create the new link on top of the old one.

The Hyperlink Manager

If you want to insert a link to external websites, an email link, or create anchors on a page, you must use the Hyperlink Manager.

With the Hyperlink Manager, you can insert a link to:

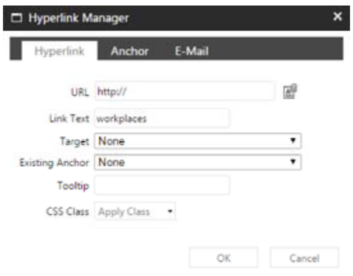
- An external website.
- A media item – for example, a video or a PDF file from your Media Library.
- An anchor in the same text field – a bookmark that links to a particular section within a page.
- An email address – a link that opens and creates a new email in the visitor's default email program.

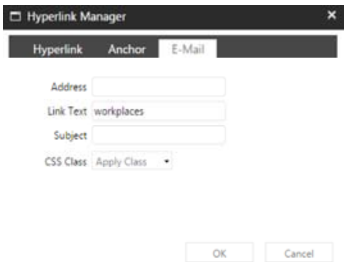
Note

The Hyper Link Manager is only available in Rich Text fields, if you want to insert links in a Word field, you must use the Hyperlink functionality from within Microsoft Word or the Insert Sitecore Link functionality.

To insert a link with the Hyperlink Manager:

1. In the Content Editor or the Experience Editor, navigate to the item or page where you want to insert a link and open the text editor.
2. In the text editor, select the text or image that you want to be a link and click Insert Link.
3. To insert a link to:
    - An external website, on the Hyperlink tab, in the URL field, enter the URL of the website that you want to link to.
    - A media item, on the Hyperlink tab, click Document Manager and select the media item that you want to link to. From the Document Manager, you can also upload a new media file to the Media Library.
    - An existing anchor, on the Hyperlink tab, in the Existing Anchor field, select the anchor that you want to link to. If the appropriate anchor is not available, you must create the new anchor first.
    - An email address, on the E-mail tab, enter the email address that you want the email sent to and enter a description in the email subject line.





4. If you want a hyperlink to have a tooltip that appears when the user moves their cursor over the link, enter this text in the Tooltip field.
5. For hyperlinks, in the Target field, specify how you want the link to open in a browser. For example, specify that you want the link to open in the same window or a new window.
6. Specify the style sheet in the CSS Class field.
7. Click OK.
8. To remove the link, select the link in the text editor and click Remove Link. Then, the link is removed and the text turns into normal text.

If you want to edit a link, you select the link in the text field and in the Hyperlink Manager dialog box, you make the necessary changes to the link and then click OK.

To change the existing link entirely, you simply create the new link on top of the old one.

Send feedback about the documentation to docsite@sitecore.net.

Create a version of an item

In the Content Editor, you can create new versions of any item in Sitecore, and you can create as many versions of the item as you like.

There are two types of versions:

- Numbered versions – versions of an item in the same language.

  For example, you can create a second version of your products page that you can use during the period of a special promotion. You can also create a new version of an image and change the new version to fit the promotion and then change it back to the original version after the promotion.

- Language versions – versions of an item in a different language.

  For example, you can create a page in Danish and then create an English version of the same page. In this case, you also want to create an additional version of any image if you want to display the Alt text in both languages.

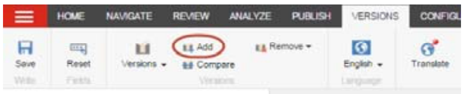
This topic outlines how to:

- Create a numbered version of an item
- Create a language version of an item
- Remove a version of an item

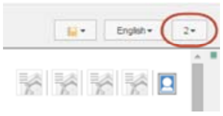
Create a numbered version of an item

To create a new numbered version of an item:

1. In the Content Editor, in the content tree, navigate to the item that you want to add a new version to.
2. Click the Versions tab and in the Versions group, click Add.

   Now in the top right corner of the editing pane of the item, you can see the item's new version number. In this example, 2.

3. Make the relevant changes to the new version and save your changes.

Create a language version of an item

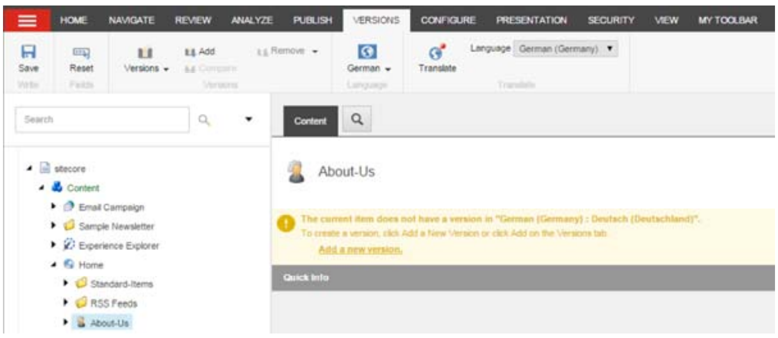
To create a new language version of an item:

1. In the Content Editor, in the content tree, navigate to the item that you want to add a language version to.
2. On the Versions tab, in the Language group, click Change the current language and in the drop-down list, select the language that you want the new version to be in.

   Note

   If the language that you want is not in the drop-down list, click More languages, and in the Select Language dialog box, click the relevant language and click OK.

3. In the notification that appears at the top of the item's editing pane, click the link Add a new version.

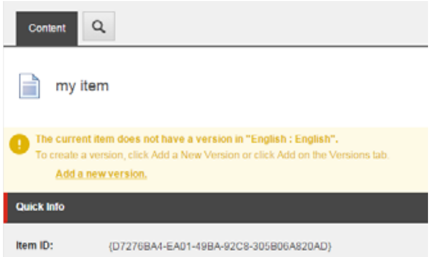
   

4. Save your changes.

Remove a version of an item

To remove a version of an item:

1. In the content tree, navigate to the item and the version that you want to remove. Make sure that you select the relevant language and numbered version.
2. In the ribbon, click the Versions tab, and in the Versions group, click Remove, to remove the selected item version.
   - To remove all the versions of the item, click the Remove drop-down arrow, and then click Remove all versions. This does not remove the item itself. Instead, a notification appears on the item, to let you know that there is no version of the item available.



Send feedback about the documentation to docsite@sitecore.net.

Explore mode

In *Explore* mode, you can simulate visits to your website to see how different personas interact with Experience Marketing features as they navigate through your website. This enables you to test your digital marketing strategy, using a variety of different visitor personas (presets) and preset modes that trigger different elements of xMarketing functionality.

In *Explore* mode you can:

- Use presets to simulate a journey through your website that collects engagement value points, converts goals, and triggers other xMarketing functionality.
- Test your profiling strategy, personalization, and other digital marketing functionality.
- Test goals and other events that you want to trigger on your website.
- Test the effectiveness of individual pages and identify potentially underperforming pages.

Note

Explore mode is designed to let you test the xMarketing functionality on your website. You cannot create or configure xMarketing functionality in *Explore* mode – you must do this in the Marketing Control Panel or in the Experience Editor.

To benefit from simulating the experience on your website in *Explore* mode, you should spend some time evaluating and assessing the xMarketing features that you have implemented on your website.

The first time you use *Explore* mode, you should keep your profiling strategy as simple as possible.

You may consider consulting the Sitecore Business Optimization Services (SBOS) team before implementing the digital marketing functionality. The SBOS team can help you develop the presets to use in *Explore* mode. Alternatively, you can consult the SBOS resources on SPN.

Send feedback about the documentation to docsite@sitecore.net.

Create an Explore mode preset

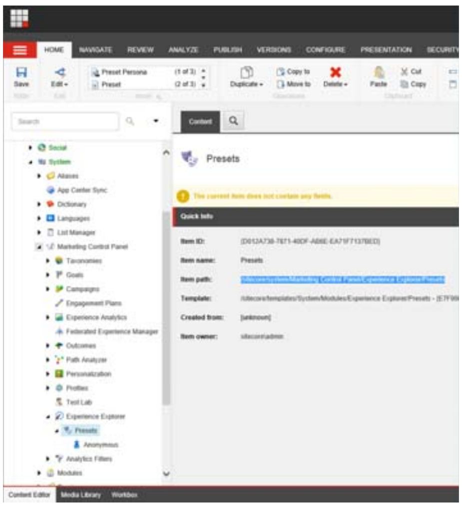
The first time you open a website in *Explore* mode there is a single preset called *Anonymous*. Creating presets is normally part of your overall Experience Marketing profiling strategy. Before you start to analyze your website in *Explore* mode, you must create your own set of presets.

For more information on creating presets or developing a content profiling strategy, see the SBOS information on SPN.

To create your own set of presets:

1. In the Content Editor, in the content tree, navigate to the */sitecore/system/Marketing Control Panel/Experience Explorer/Presets* node.

   You should only see one default preset called *Anonymous*.

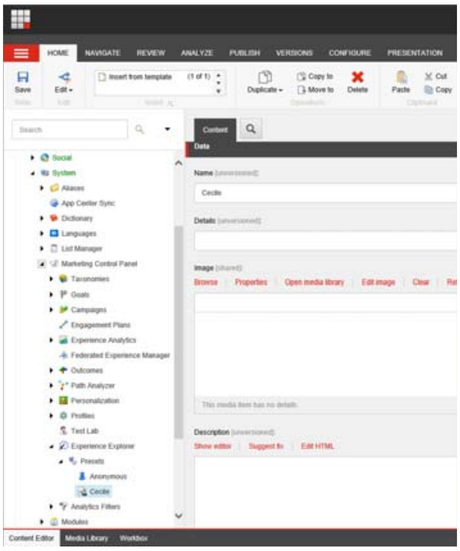
   

2. In the ribbon, click Home, Insert, and then click *Preset Persona* and give the preset a name - *Cecile*.

   You can create two types of item:

- *Preset – Persona*
- *Preset*

The *Preset - Persona* item contains more fields that you can use to describe your preset than the *Preset* item.

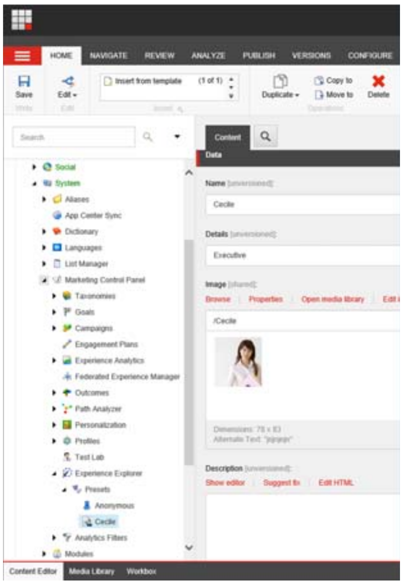


3. Enter a name and any other relevant details.

You should enter enough information to create an accurate portrayal of your preset. The most important fields to include are:

Field name	Description
Name	Name of the preset. For example, 'Cecile'.
Details	Type of business visitor. For example, 'Executive'
Image	Attach a suitable image to represent your preset.
Description	Provide a detailed description of the type of business visitor this preset represents. For example, 'Cecile is a company CEO looking for first class travel and a high standard of business accommodation'.

You can leave the other fields empty for now.



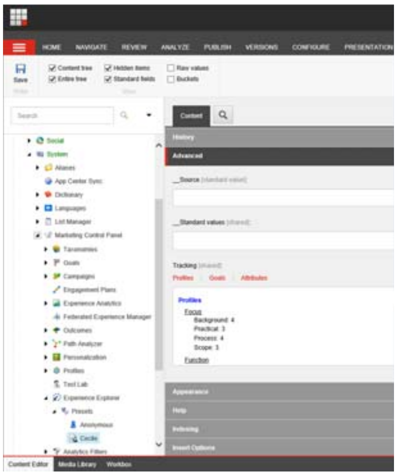
4. Create the other presets that you have prepared as part of your profiling strategy.
5. Save your changes.

Configuring the preset

After you create a preset, you can associate it with some specific Experience Marketing functionality.

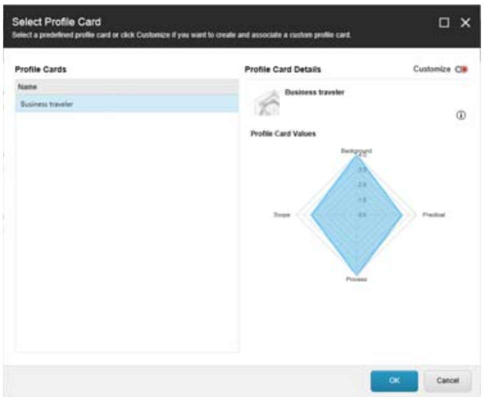
To configure a preset:

1. In the Content Editor, navigate to the */sitecore/system/Marketing Control Panel/Experience Explorer/Presets* node and select the preset that you want to configure.
2. In the Advanced group, you can see the profiles, goals, and campaigns associated with this preset.



3. Click on Profiles, Goals, or Attributes to edit any of these values.

   For example, click Profiles to view or edit the profile card values assigned to the *Cecile* preset.

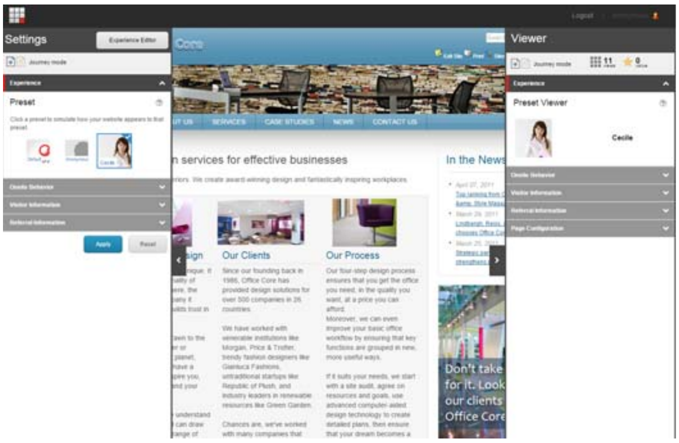


4. Click Customize to change these values.
5. Click OK to save your changes.

Seeing the presets in Explore mode

Open the website in *Explore* mode and in the Settings panel, click the Experience section to see your new preset displayed in the Preset carousel.

Select a preset and then click Apply.



Send feedback about the documentation to docsite@sitecore.net.

Explore mode - a sample visit

You use *Explore* mode to simulate the experience of visiting your website from the perspective of a real visitor. When you select a preset, the look and feel of your website is adapted during the visit to match the experience of one of your important business target groups specified by the corresponding preset.

- In an *Explore* mode – *Fixed* visit, as the visitor navigates through your website, all their profile values are automatically reset to the values specified in the selected preset. In a *Fixed* visit, each page of the website appears to the visitor as if they landed directly on that page as the first page of their visit or kept their profile fixed.

  This means that any goals, campaigns, or profile values that are triggered during the visit are automatically reset and are not accumulated as values during the visit.

- In an *Explore* mode – *Journey* visit, as the visitor navigates through your website, all the values are accumulated and these values follow the visitor so that the visit simulates a real journey.

  This means that any goals, campaigns, or profile values that are triggered during the visit are automatically associated with the visitor and are accumulated during the visit.

Presets

Presets are personas that have values associated with them such as goals, campaigns, profile values, or conditional renderings. The SBOS team can help you to develop the appropriate presets to use with your own website.

This example uses a preset:

- Cecile - a company CEO who travels first class and expects a high standard of business accommodation.

The scenario simulates the experience that Cecile has when she visits different pages of the Jetstream website. As she navigate through the website, you can see how the website adapts and changes according to her movements, as she achieves goals, changes her behavior, and visits different pages on the site.

This scenario also describes the difference between a *Fixed* visit and a *Journey* visit.

Selecting a preset

Before you can simulate a visit to the website, you must select the appropriate preset.

To select a preset and the Explore mode:

1. In the Experience Editor, in the Mode group, click Other, Explore.
2. In the Settings panel, expand the Experience section and select the relevant preset, for example, *Cecile*.
3. At the top of the panel, select the Explore mode - Fixed or Journey.
4. Click Apply.

Note

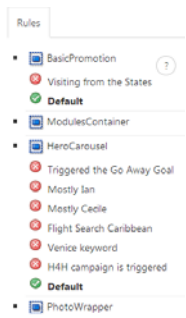
When you click Apply, this does not actually save any changes to the values defined for the preset. You can only make permanent changes in the Content Editor.

You can always reset the profile values and other xMarketing values of the presets to their initial state.

The Home page

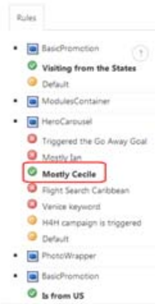
On the Jetstream website, the default *Home* page displays information about Venice:

In the Viewer panel, in the Page Configuration section, you can see that in the *Hero Carousel* spot, no personalization rules have been triggered.



When Cecile visits the website, the *Home* page displays information about Paris because this is specified by the *Mostly Cecile* personalization rule.

In the Viewer panel, in the Page Configuration section for both a *Fixed* and a *Journey* visit, you can see that in the *Hero Carousel* spot, the *Mostly Cecile* personalization rule has been applied and is displayed in bold with a green icon. This happens because the *Cecile* preset values closely match the values specified for this personalization rule (*Mostly Cecile*).



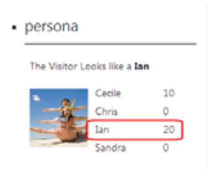
The Destinations page

Cecile is curious about possible destinations and navigates to the *Destinations* page.

She clicks the *Caribbean* destination.

The Caribbean is a popular vacation destination for families and is therefore a better profile match for the more family-oriented profile of the Ian preset. Ian is a family man looking for a holiday destination that is all-inclusive, offers a wide variety of fun activities for the family, and provides reliable childcare facilities.

In a *Fixed* visit, when Cecile visits the *Caribbean* page, 20 points are added to the Ian persona profile key for Cecile. This is because the Ian persona has become the most dominant.



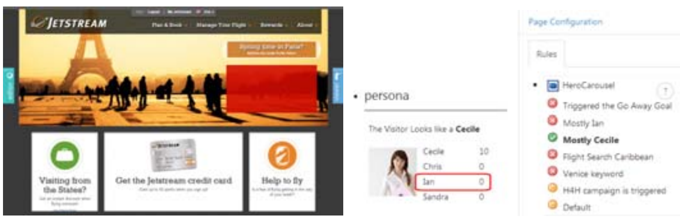
This value is only visible when she views this page – when she leaves the page, the values for the Cecile preset are reset to 0.

In a *Journey* visit, when Cecile visits the *Caribbean* page, it is personalized for the Ian persona profile key and the value points are kept.

Back on the Home page

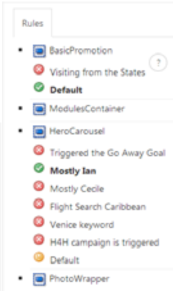
Cecile returns to the *Home* page.

In a *Fixed* visit, she still sees the front page about Paris because in a *Fixed* visit, all the values are automatically reset and match the selected preset on each page that is requested. The profile value of 20 points that was added to Ian's profile is reset to zero; the profile value that is specified in the Cecile preset for this profile is added instead.



In a *Journey* visit, the *Home* page is now personalized for the Ian profile because Cecile's chose a family destination.

The Ian profile is now the most dominant profile for this visit – it has 20 points while the Cecile profile only has 10 points. The home page therefore displays information about a family holiday destination instead of a typical business trip destination.



In the Hero Carousel, *Mostly Ian* has replaced *Mostly Cecile* as the applied personalization rule.

Resetting your profile

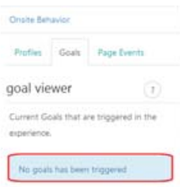
If at any stage you want to alter the values and profile characteristics, you can make these changes directly in the Settings panel by expanding the Onsite Behavior section and in the *Persona* profile changing the values. The values are updated when you click Apply.

For example, you can change the value of the *Ian* profile key to zero or 50.

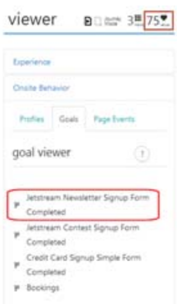
The Newsletter Signup page

Cecile decides to sign up for the Jetstream newsletter. When she submits the form, it triggers the *Jetstream newsletter signup form* goal and adds 5 engagement value points to her visit.

In a *Fixed* visit, the goal that is applied to the submit form button is not triggered.



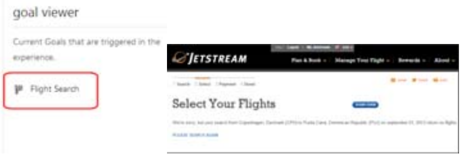
In a *Journey* visit, in the Onsite Behavior section, you can see that the goal is registered and Cecile gains 5 engagement value points. The Cecile preset has been configured with several goals that she has already achieved, giving her an additional 70 engagement value points. After she has signed up for the newsletter, she has a total of 75 engagement value points.



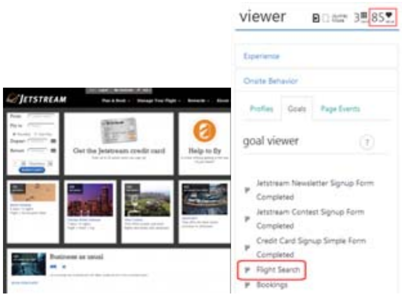
Searching

Cecile decides to search for a flight from Copenhagen to Punta Cana in the Caribbean.

In a *Fixed* visit, when she is on the *Search results* page, no other triggered goals are displayed; only goals that are triggered on the current page are shown. Cecile's search brings up no results, but her search triggers the *Flight Search* goal.



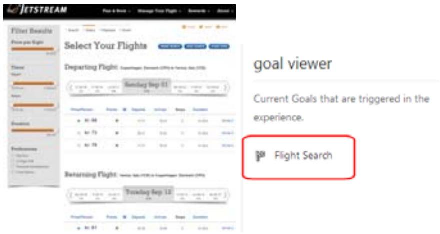
In a *Journey* visit, the *Flight Search* goal adds another 10 engagement value points to her profile.



No results were found, so Cecile decides to search for another flight, this time from Copenhagen to Venice. Performing this search adds another 10 engagement value points.

The Select Your Flights page

This time, her search returns several flights. She selects a suitable flight and books it. On this page, she has triggered the *Flight Search* goal again.



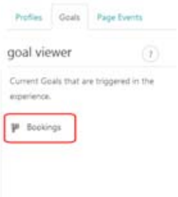
In a *Fixed* visit, no other triggered goals are displayed.

In a *Journey* visit, booking a flight adds 70 engagement value points giving Cecile's profile a total of 165 points.

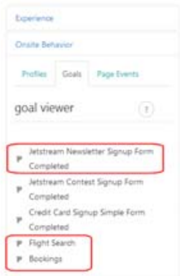
The Thank You page

On the *Thank You* page, Cecile triggered the *Bookings* goal.

In a *Fixed* visit, because only goals that are triggered on the current page are shown, the *Bookings* goal is displayed and the *Flight Search* goal is no longer visible.



However, in a *Journey* visit, the Viewer panel for her journey shows that Cecile has also achieved 3 extra goals during this visit.



Send feedback about the documentation to docsite@sitecore.net.

The Explore mode settings

In *Explore* mode, the Settings panel is on the left side of the webpage.

The header displays the type of visit that you have selected for the visitor experience:

- Journey mode – the preset values are the starting point for your visit. As you travel through the website, you accumulate engagement value points and you can monitor any changes that are made to the site, for example, through personalization.
- Fixed mode – the preset values are permanent and will not change as you travel through the website. If you visit a page that contains personalization, you see the page change but all the changes are cleared when you move on to another page. You do not accumulate any engagement points during a *Fixed* visit.

Note

When you are in *Explore* mode, the Experience Editor ribbon is disabled. To leave *Explore* mode and enable the Experience Editor ribbon, in the Settings panel, click Experience Editor.

The following table describes the options that are available in the Settings panel.

Note

The *Explore* mode options described in this table can vary depending on how you have implemented your website. For example, when you install Sitecore, the only presets that are available are *Default* and *Anonymous*.

Section	Setting	Options	Description
Experience	Preset	• Default • Anonymous • The presets you create	Select the preset that you want to use as you travel through the website. Each preset consists of a persona and some other settings that enable you to experience the website from the perspective of different types of visitors.

Onsite Behavior	Profiles	The profile keys that you created for the website. For example: • Visit Profile • Business • Couples • Duration • Family • Repeat visits • Tourism	You can add values to the existing profile keys for this visit. Note You must create and configure profile values for each profile in the Marketing Control Panel.
	Goals	A list of goals, for example: • Registration form completed • Frequent flyer form completed • Facebook clicked	Select the goals that you would like to trigger during the visit.
	Page Events	A list of page events, for example: • Campaign • Download • Print • RSS	Select the page events that you would like to trigger during the visit.
Visitor Information	Device	A list of devices: • Default • Print • Feed • Mobile • Tablet	Select the device that you want to use for the visit. The layout of the website changes depending on the specific device you use. For example, to experience the website from the perspective of a mobile visitor, you can select a Mobile device.
	GEOIP	Map	Enter the GEOIP information that you want to use for this visit and click Apply. This information can be used for a MaxMind lookup. Click on the map to specify your location for the visit or enter the appropriate latitude and longitude.
		Country	Select the country that you want to use for the visit.
		IP	Enter an IP address that you want to use for the visit. Important If you select more than one option, they could be in conflict. The Viewer panel therefore only displays the last option that you selected before you clicked Apply.
Referral Information	Campaigns	A list of campaigns, for example: • Frequent flyer promotion campaign • Home for Holidays campaign	Search for or select the campaign that you would like to trigger during the visit.
	Referral	Referral requests field.	Enter the URL of a website that links to your site. Alternatively, you can enter a keyword from a search engine such as Google or Bing.

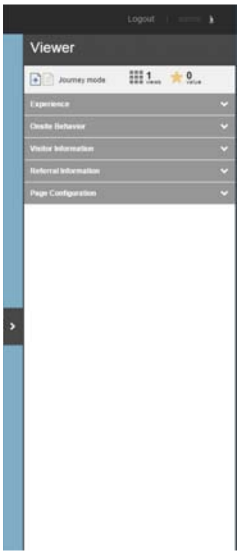
Note

If you want to change the order of the sections in the Settings panel or in the Viewer panel, you should contact a developer or administrator. You can also use role-based permissions to hide or show specific sections in the panels. For example, you can create a dedicated Explore mode Marketer role that only has access to Geo IP and Campaigns.

Send feedback about the documentation to docsite@sitecore.net.

The Explore mode Viewer panel

In *Explore* mode, the Viewer panel is on the right side of the webpage.



The header in the Viewer panel displays:

- The mode that you selected for the visitor experience. There are two options *Journey* or *Fixed* mode.
- The current total number of page views.
- The current total of the engagement value points that you have accumulated during your visit.

The following table describes the information that is available in the Viewer panel.

Note

The *Explore* mode options described in this table can vary depending on how you have implemented your website. For example, when you install Sitecore, the only presets that are available are *Default* and *Anonymous*.

Section	Setting	Description
Experience	Preset	Displays the name of the preset that you have selected.
Onsite Behavior	Profiles	Depending on the profiling strategy that you have implemented, you can see which profile pattern card most closely matches the pages visited by the preset.
	Goals	Displays the goals triggered during the visit. The goals can be either accumulated in *Journey* mode or on the page in *Fixed* mode.
	Page Events	Displays the page events that have been triggered during this visit.
Visitor Information	Device	Displays the device that you have selected to use for this visit to the website.
	Geo IP	Displays a Google map, a country, or IP information depending on which was last updated in the Visitor Information section in the Settings panel.
	Tags	Displays the current tags that are associated with the visitor.
Referral Information	Campaigns	Displays the campaigns that are associated with this visit.
	Referral	Displays the referred sites or search keywords that are associated with this visit.
Page Configuration	Rules and conditional renderings	Shows the rules and conditional renderings that are present and active on the current page. Holding the mouse over a rule shows how the rule should be evaluated or how it should be evaluated if the preceding rule is not true.

Note

If you want to change the order of the sections in the Settings panel or in the Viewer panel, you should contact a developer or administrator. You can also use role-based permissions to hide or show specific sections in the Settings panel or in the Viewer panel. For example, you can create a dedicated Explore mode Marketer role that only has access to Geo IP and Campaigns.

Send feedback about the documentation to docsite@sitecore.net.

Items

Items are the basic building blocks of your Sitecore website. An item can represent any kind of information that makes up a webpage, for example, a piece of text, a media file, a layout etc.

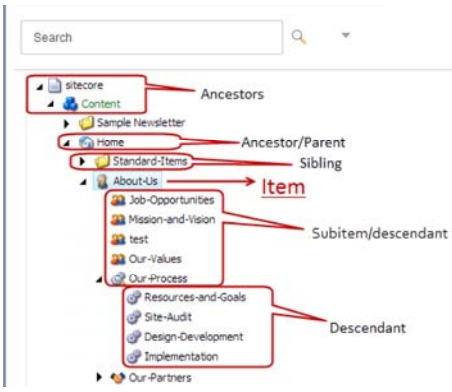
An item always has a name, an ID that uniquely identifies the item, and it is based on a template that defines which fields the item contains. In addition, an item can have multiple versions and multiple language versions.

You can use the Content Editor or the Experience Editor to edit, create, and insert items.

- In the Content Editor, all the items on your website are stored in the content tree, where you can navigate to a specific item in a hierarchy.

  In the content tree, the relationships between an item and its associated items are as follows:

  - Parent – the item directly above the item
  - Sibling – an item at the same level as the item
  - Subitem – an item directly below the item.
  - Ancestors – all the items that are above the item
  - Descendants – all the items that descend from the item

- In the Experience Editor, the items are displayed as they look on your website. The Experience Editor is a WYSIWYG (What You See Is What You Get) editor that allows you to edit items directly on the page.

  Note

  The ways in which you can create and edit items in Sitecore can vary depending on your security settings and the way your website is setup.

In both the Content Editor and the Experience Editor, an item must be in the editing state before you can edit it. You can lock an item, before you start to work on it to ensure that no other content authors work on the item at the same time.

Send feedback about the documentation to docsite@sitecore.net.

Media items

A media item is composed of several fields and it always has a file attached to the Media field. A media item can be an image, a document, a video, or an audio file.

In Sitecore, there is a difference between media files and media items. An image or video file that is stored on your computer is called a media file. When you upload the media file to the Sitecore Media Library, the file is attached to a media item and is from then on referred to as a media item.

A Media item can be:

- Images — .jpg, .gif, .png and other formats.
- Microsoft Word documents — .doc and .docx files.
- PDF documents
- Videos — .MPEG, .WMV and other formats.
- Audio files — .MP3 and other audio formats.

You should store and manage your media items in the Media Library. To edit your media items, you can use the Sitecore tool the Image Editor for images or other applications that you have installed on your computer such as Photoshop or Microsoft Word for documents. You can open the Image Editor from the Sitecore Start menu.

Send feedback about the documentation to docsite@sitecore.net.

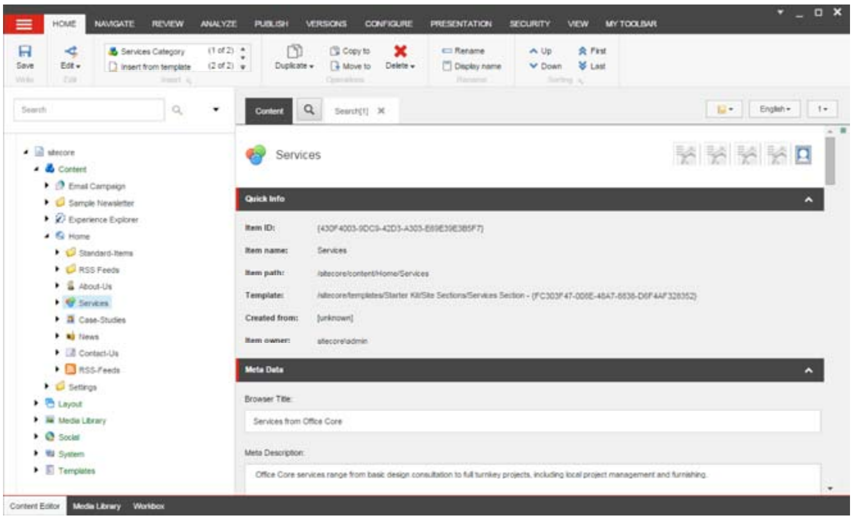
The editing tools

In Sitecore, there are two editing tools where you can create and edit the content on your website:

- The Content Editor – an application that is designed for more experienced content authors who are familiar with Sitecore and the functionality that it contains.
- The Experience Editor – an intuitive editing environment that is designed for content editors who want to edit and write content directly on the page.

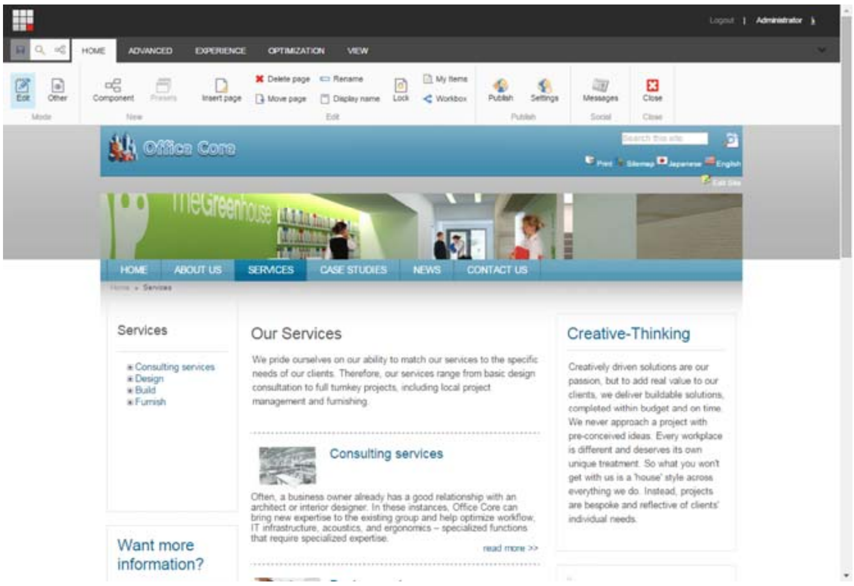
In the Content Editor and the Experience Editor, the items that make up your website are presented in very different ways. In the Content Editor, the items are objects in the content tree and when you select an item, you can edit its fields. In the Experience Editor, the items are presented as they look on the website and you can edit them directly on the page.

For example, in the Content Editor, the Services item of a sample website looks like this:



When you click the Services item in the content tree, the Content tab displays the Services item's field values, which you can also edit. If you want to edit another item, you must click the item in the content tree.

In the Experience Editor, you see the Services item presented as it looks on your website:



You can edit all of the items on the page – text, graphics, logos, links, and so on, even if they do not belong to the selected item.

Send feedback about the documentation to docsite@sitecore.net.

Copy or duplicate an item

In the Content Editor, you can create a new item by copying or duplicating an existing item:

- Copy an item – copies the selected item and all its content and subitems to a location in the content tree that you specify.
- Duplicate an item – copies the selected item and all its content and subitems to the same location in the content tree.

Note

In Internet Explorer, you can also copy, cut, and paste items to and from the clipboard. The clipboard functionalities are available on the Home tab, in the Clipboard group, but due to JavaScript restrictions in other browsers, the Clipboard group appears only in Internet Explorer.

Copy an item

To copy an item to another location:

1. In the content tree, locate and select the item that you want to copy.
2. On the Home tab, in the Operations group, click Copy to. Alternatively, you can also right-click the item in the content tree, click Copying and then Copy to.
3. In the Copy Item To dialog box, select the location where you want the copy the item to be stored.

   cms_ce_CopyItemTo_Dialog

   Note

   You can navigate to the location through the content tree on the Browse tab, or you can search for the location on the Search tab.
4. Click Copy to create a copy of the item in the specified location.

Now you can edit the new item, add subitems, or move the item to another location.

Note

To copy multiple items at the same time, you must run a search for all the items that you want to copy and use the *Copy results to* search operation.

Duplicate an item

To duplicate an item:

1. In the content tree, locate and select the item that you want to duplicate.
2. On the Home tab, in the Operations group, click Duplicate. Alternatively, you can right-click the item and click Duplicate.
3. Enter a name for the new item and click OK.

Now you can edit the new item, add subitems, or move the item to another location.

Send feedback about the documentation to docsite@sitecore.net.

Create an item in the Content Editor

When you create a new item in the Content Editor, you must first select an item in the content tree to indicate where you want the new item to be located. You can create a new item as a sibling or a subitem to the item that you select:

- A sibling is an item that you create at the same level as the selected item.
- A subitem is an item that you create under the selected item.

This topic outlines how to

- Insert a new item
- Insert an item based on a template

  Tip

  You can also create new items by copying or duplicating existing items.

Insert an item

To insert a new item:

1. In the Content Editor, navigate to the item where you want your new item to be located.
   - To insert a sibling to the selected item, on the Home tab, in the Insert group, click the drop-down arrow and in the Insert a new sibling section, select the item type that you want to insert.

cms_ce_HomeTab_InsertGroup_DropDownArrow

- To insert a subitem to the selected item, on the Home tab, in the Insert group, use the up and down arrows to look for and select the type you want to use, enter the name of the new item, and click OK.

  cms_ce_HomeTab_InsertGroup_ScrollButtons

   Alternatively, you can right-click the item in the content tree, click Insert, and then click one of the suggested item types.

2. Enter a name for the new item and click OK.

   Every new item that you create automatically locks and opens in the Content Editor. You must unlock and save your items before other users can work on them.

3. As appropriate, edit the new item, add subitems, or move your item to another location.

Insert an item based on a template

If you have sufficient access rights, you can create a new subitem from any template that is available in your Sitecore installation.

To insert an item from a template:

1. In the content tree, find and select the item that you want to create a subitem for.
2. On the Home tab, in the Insert group, click the drop-down arrow and click Insert from template.
3. In the Insert from Template dialog box, navigate to the template that you want to base the new item on.

   cms_ce_InsertFromTemplate_Dialog

   Alternatively, you can right-click the item in the content tree, click Insert, and then click Insert from template.

   Note

   To locate the template you can navigate through the content tree on the Browse tab, or you can use the Search tab to search for the item.

4. In the Item name field, enter a name for the new item and click Insert.
5. As appropriate, edit the new item, add subitems, or move it to another location.

Important

You can create an item in an item bucket in exactly the same way as you create a regular item. If the template that the new item is based on is bucketable or if you make the item bucketable, the item is hidden in the item bucket. If the item or the item's template is not bucketable, the item is visible in the item bucket. Consequently, you can have an item bucket that is a hybrid of hidden items and visible items..

Send feedback about the documentation to docsite@sitecore.net.

Create an item in the Experience Editor

You can create new items and insert them directly on the webpage using the Experience Editor.

Note

How you create items in Sitecore can vary depending on the security roles that you have and the way your website is set up.

To insert a new item:

1. In the Experience Editor, navigate to the page that you want to add a new item to.
2. In the ribbon, on the Home tab, in the Edit group, click Insert page.
3. In the Insert Content dialog box, select the type of item that you want to insert, and in the Content item name field, enter a name for the item.

   cms_xe_InsertContentDialog

   The type of item that you can create varies depending on where you are on the website. For example, if you are on the front page, you can insert a new section and you can add appropriate items to that section.

4. Click OK. The new section containing no items appears on the website and you can now edit the item by adding text, images, sidebar text, and so on.

   Note

   When you create a new item, it is automatically locked to prevent other users from editing it.

5. When you are finished editing the new item, click Save and then Unlock to make the item available to other content authors.

Send feedback about the documentation to docsite@sitecore.net.

Edit content directly on the page

The Experience Editor is a WYSIWYG editor that allows you to easily make changes to items directly on the page. You can edit all the items that are visible on the page — text, graphics, logos, links, and so on.

The Experience Editor, you can edit the following field types directly on the page:

- Simple text fields – this field can only contain simple text, such as single-line and multi-line text, dates, and links.
- Rich Text fields – this field gives you more control over how the text appears on the webpage. You can edit a Rich Text field directly on the page or open the Rich Text Editor to get access to more formatting functionality.
- Word fields – this field allows you to use Microsoft Word to edit the text. Depending on the configuration of your website, you can edit the Word field in Microsoft Word or directly on the page. If your website supports inline editing, all of the features of Microsoft Word are also available in the inline editing pane.
- Image fields – a field that only contains images. In an image field, you can change the image, modify the image appearance, or remove the image directly on the page.

This topic outlines how to:

- Customize the Experience Editor view
- Edit a field directly on the page

Customize the Experience Editor view

If you have the appropriate access rights, you can change the way the Experience Editor presents the editable items for you.

On the View tab, you can setup the Experience Editor:

- To show the editable text fields, select the Editing check box. When you move your mouse over the fields in the Experience Editor, the text fields appear with a dotted line.
- To show the renderings and placeholders, select the Designing check box. When you move your mouse over the fields in the Experience Editor, the renderings and placeholders appear with a dotted line.
- To show outlines of all the objects on a page, select the Controls check box. Use this feature to understand how the page is organized. This check box is only active when the Designing check box is selected.

Edit a field directly on the page

To edit a field directly on the page:

1. Log in to the Experience Editor and navigate to the item that you want to edit.
2. Click the item and enter the appropriate text. Depending on which type of field you select, you have different options in the floating toolbar that appears:
    - In a simple text field, you can only enter text. The field automatically expands to fit the text you enter.

      cms_xe_SimpleTextField_FloatingToolBar

    - In a rich text field, you can perform simple formatting on the text and insert an image or a link.

      cms_xe_RichTextField_FloatingToolbar

      Note

      The options available in the floating toolbar vary depending on your version of Sitecore CMS and your access rights.

    - In a Word field, you can insert an image or a Sitecore link, and choose how you want the Word ribbon displayed:

      cms_all_WordEditor

    - In an image field, you can insert various types of media. These could be animations, film clips, sound files, and so on.

      cms_xe_ImageField_FloatingToolbar

3. If you want to edit the item's field values in the Content Editor, click the More drop-down arrow and click Edit the related item. This opens the item in the Content Editor above the Experience Editor in the same browser.

    To return to the Experience Editor, click the Sitecore Start button and select Exit.

4. When you have finished editing the field, click Save to save your changes.

Edit the website content

When you edit content in rich text fields, you can benefit from the advanced editing functionalities in the Rich Text Editor or in Microsoft Word. For minimal editing functionality, you can edit rich text fields directly on the item or page in the Experience Editor.

The Rich Text Editor

The Rich Text Editor is the standard text editor in Sitecore. In this editor, you can edit all the content in a rich text field.

In the Rich Text Editor, you can:

- Paste text from Microsoft Word.
- Find and replace text.
- Insert pictures or other media items.
- Insert and delete links to other parts of the website and to external websites.
- Insert symbols.

Note

The Rich Text Editor is a standard word processing tool, but in Sitecore, most of the formatting functionality has been removed because the style sheets and templates for a website determine the format of the items that content authors create and edit.

Microsoft Word

You can use Microsoft Word as your text editor in Sitecore and benefit from all the functionality that is available in Microsoft Word. You can open the Word field editor from both the Content Editor and the Experience Editor.

- Insert a link to an item in Sitecore.
- Insert an image from the Media Library.

Important

To edit a text field in the Word field editor, your administrator must set up the text field as a Word field. In addition, make sure to use Microsoft Word 2007 or later and Internet Explorer 7 or later. If you use Internet Explorer 8, you must configure your security settings to support Word fields in Sitecore.

Edit a text field

To edit a text field in the Content Editor or in the Experience Editor:

1. In the Content Editor or the Experience Editor, navigate to the item or page that you want to edit. Make sure to select the relevant version and language of the item.
2. To open the text editor:
   - From the Content Editor – in the content area, locate the Rich Text field you want to edit and click Show Editor. If the text field is a Word field, you click Edit.
   - From the Experience Editor – click the field that you want to edit and in the floating toolbar that appears, click Edit the text Edit" height="12" width="12" alt="button_xe_FloatingToolbar_EditText" title="Edit the text button" class="documentimage" doc-fancy-image="">. If the text field is a Word field, click Edit the word document .
3. Make the appropriate changes to the content in the text field.
   - In the Rich Text Editor, if you prefer to edit the content in HTML format, click HTML.

     cms_RTE_FullRibbon

     Note

     Your Sitecore system administrator controls the features that are available to you. If you need more features, ask your administrator to make them available.
   - In a Word field, to insert a Sitecore link to another item on your website, click Insert a link. To insert an image from the Media Library, click Insert media item. If the image is not available in the Media Library, you can use the insert picture functionality in Microsoft Word.


cms_all_WordEditor

Note

Images that you want to use regularly, you should upload to the Media Library.

4. When you are finished editing the content, click Accept to save your changes.

Note

To make sure that links do not break when items are, for example, renamed, deleted, or moved, all the links to Sitecore items or media items should have the format:

```
<img ... src="-/media/{ITEM_ID}.ashx" />
```

or

```
<a href="~/link.aspx?_id={ITEM_ID}&amp;_z=z">
```

For example, `<img alt="logo" height="51" width="204" src="-/media/094AED0302E7486880CB19926661FB77.ashx" />`

Send feedback about the documentation to docsite@sitecore.net.

Lock and unlock an item

In Sitecore, you can lock the item you are working on to prevent other users from editing the same item at the same time. If the item that you want to edit is already locked, you cannot edit it, you can only view it. When you have finished editing the item, you must unlock the item to make it available to other users again.

Depending on the set up of your Sitecore installation, you may be prompted to lock the item, before you can edit it.

A Sitecore administrator can have the authority to edit another user's locked items. Therefore, an administrator and another user can edit an item at the same time. If a user saves changes to an item, and another user saves changes to the same item afterwards, a message box appears, asking the user if the previously saved changes should be overwritten.

Important

Locking an item does not keep it from being published. If you are working on an item and you want to keep it from being published, you need to set up the publishing restrictions for the item.

This topic outlines how to:

- Lock an item
- Unlock an item

Lock an item

To lock an item manually:

1. In the Content Editor or the Experience Editor, navigate to the item or page that you want to edit.
   - In the Content Editor, on the Home tab, click Edit to lock the item.

     
     cms_ce_FullRibbon_EditButton

     Note

     In the content tree, you can use the Quick actions to see a quick overview of the items that you have locked. Right-click left of the content tree and select My Locked Items.

   - In the Experience Editor, on the Home tab, in the Edit group, click Lock.

cms_xe_Ribbon_LockButton

2. When you are finished editing the item, make sure to unlock the item again.

Unlock an item

If you have locked and edited numerous items, you can lose track of them. Sitecore keeps track of all the items that are locked, and by who. In this way, you can easily lock one or more items at the same time.

To unlock an item

1. To see a list of the items that are locked by you, or to unlock one or more items:
   - In the Content Editor, on the Review tab, in the Locks group, click My Items.

     cms_ce_ReviewTab_LocksGroup_MyItems

   - In the Experience Editor, on the Home tab, in the Edit group, click My Items.

     cms_ce_ReviewTab_LocksGroup_MyItems

     Note

     The number next to the My Items button on the ribbon indicates the number of items that you have locked.

2. In the My Items dialog box, select the items that you want to unlock and click Unlock. To unlock all items at the same time, click Unlock All.

   cms_ce_Ribbon_ViewTab_MyItemsDialog

Send feedback about the documentation to docsite@sitecore.net.

Preview a webpage

The preview functionality lets you preview all your items on your website without publishing them first.

The preview functionality lets you:

- Specify a preview date, so that you can view your website in the past or future, according to the publishing restrictions that are set up.
- See how a webpage looks in different language versions.
- Preview how the page displays on different devices.

This topic outlines how to:

- Preview a page on your website
- Preview a page in the Device Simulator

Preview a page on your website

You can open the preview mode from either the Content Editor or the Experience Editor.

To preview a webpage:

1. To open the preview:
   - From the Experience Editor, on the ribbon, in the Mode group, click Preview.

- From the Content Editor, on the Publish tab, in the Publish section, click Preview.

2. On the Experience tab, select the preview parameters:
   - Date – to specify another date to preview the webpage, in the Date group, click the date, or click the blue arrows to select the previous day or the next day.
   - Language – to view the page in a different language, in the Language group, click Language .

   cms_xe_ExperienceTab

   Note

   The Experience tab might not be available to you if you do not have the necessary security permissions.

3. Navigate through the pages to preview your whole website.

   Tip

   In the preview mode, you can also use the search functionality to navigate to a specific page that you want to preview.

Preview a page in the Device Simulator

The Device Simulator enables you to preview your website in a simulation of several different devices, such as an iPhone, iPad, Android Phone, and Blackberry.

The Device Simulator mimics the devices and the preview is not a pixel-perfect rendition of the website as it appears on each device. However, it gives you a good indication of how the device presents the website and it lets content authors quickly see how pages look on mobile devices.

To preview a page in the Device Simulator:

1. In the Experience Editor, on the Experience tab, in the Mode group, click Preview.
2. In the Simulator group, click the simulator button and select a device.

   cms_xe_ExperienceTab_SimulateDropDownMenu

3. In the Device Simulator, all the navigation remains fully functional. You can:
   - Browse through the website.
   - Click Rotate to change the orientation of the device.
   - Scroll up and down, as well as left and right.

   cms_xe_TheDeviceSimulator

4. To return to the default page preview, click the simulator button again and in the drop-down menu, click None.

Picture 7

Send feedback about the documentation to docsite@sitecore.net.

Publishing

When you edit or create new items in Sitecore, they must be published before they appear on your website. This lets you save unfinished items and decide exactly when the items are launched to the website.

When you work in Sitecore, the items you save are saved to the Master database. When you publish new items or updated items, the items are copied from the Master database to a publishing target, and from the publishing target, the website is launched. The default publishing target in Sitecore is the Web database.

You can choose to publish the entire website or a single item:

- Site publishing – publishes your entire website starting from the root of the content tree. You can publish a site from the Content Editor or from the Sitecore Desktop.
- Item publishing – publishes the item you select in either the Content Editor or the Experience Editor. The item can only be published if all its ancestors have been published. With item publishing, you can choose to include all of the selected item’s subitems and related items.

Send feedback about the documentation to docsite@sitecore.net.

Navigate to an item in the Experience Editor

In the Experience Editor, you can navigate to the page or item that you want to work on using one of the following methods:

- Preview mode – use this method if you want to browse the website to get to the page.
- Navigation bar – use this method if you want to navigate to the page in a menu structure.
- Search – use this method if you want to search for the specific item. This is very helpful if you need to work on an item or page on another website.

Preview mode

If you enable the preview mode, you can browse to a specific page like on a regular website.

- To enable the preview mode, in the ribbon, in the Mode group, click Other, and in the drop-down menu, click Preview.

  cms_xe_HomeTab_ModeGroup_Other

  You can now click the items on your website to browse to the relevant page.

Navigation bar

The Navigation bar allows you to navigate to a specific page through a menu structure.

- To enable the Navigation bar, in the ribbon, on the View tab, select Navigation bar.

  cms_xe_ViewTab_ShowGroup_NavigationBarCheckbox

  The Navigation bar appears below the ribbon.

  cms_xe_ViewTab_ShowGroup_NavigationBar

- To navigate to a page, click the item or the arrow in the Navigation bar to navigate to a page in the menu. If you want to edit the item’s field values in the Content Editor, click Edit .

Search

In the Experience Editor, you can use search to navigate to a specific item and from the search results direct the Experience Editor to the specific page where the item is located. In addition, if you work on multiple websites, you can use search to quickly move from one website to another.

- To use search to redirect the Experience Editor to a specific item, click Search in the Experience Editor ribbon.

cms_xe_Ribbon_SearchButton

In the Navigate to Item dialog box, you can search for an item across all your websites and direct the Experience Editor to that specific item.

cms_xe_NavigateToItem_Dialog

The Content Editor

The Content Editor is an editing tool that you can use to manage and edit all the content on your website. It is designed for more experienced content authors who are familiar with Sitecore and the functionality that it contains.

The Content Editors appearance and functionality vary depending on the user's roles, the local security settings, and the customizations that have been implemented on the Sitecore installation.

The user interface

The Content Editor's user interface consists of three main areas that you can customize to fit your individual needs when you work in the Content Editor. The three areas are:

- The ribbon – the area where all the functionality is available.
- The content tree – the area where all the items are organized.
- The content area – the area where you can edit your items.

cms_ce_TheContentEditor

The ribbon

The ribbon consists of the System menu and tabs where all the functionality that you can use in the Content Editor is available.

Depending on your access rights and security settings, the functionalities that are available to you in the ribbon can vary. As a minimum, you can expect to see:

cms_ce_TheContentEditor

If you have more extensive access rights, you have access to more commands and more tabs. If you have complete access to the Content Editor or if you are a Sitecore administrator, you will see something like this:

cms_ce_FullRibbon

The content tree

One of the main features of the Content Editor is the content tree, which presents all the items that are created in your Sitecore installation.

cms_ce_ContentTre
e

The Sitecore content tree has a similar structure to the one used in Windows Explorer, however, the Sitecore objects and groups are not represented by standard folder and file icons but by many different icons.

You can expand and open every group in the Sitecore content tree just as you can with the folders in Windows Explorer. When there is a plus '+' beside an icon, it contains subitems and can be expanded. When there is a minus '-' beside an icon, it has already been expanded and when there is no symbol, the item contains no subitems.

The content area

The content area is where you edit an item. When you click an item in the content tree, two tabs appear in the content area:

- The Content tab, where you can edit any of the item's field values.

  Note

  If you do not see a field as expected, on the ribbon, click the View tab, and then select the Standard Fields check box. This shows all the fields from the Standard template on every item that you open.

- The Search tab, where you can search through the selected item.

In the top-right corner of the content area, additional functionality is available that is specific to the item that is selected.

cms_ce_ContentAreaFunctionalities

The functionalities are:

- Navigate – navigate to a specific section or field in the list of field values.
- Language – toggle between the language versions of the item or to create a new language version.
- Versions – toggle between versions of the item or to create a new version.
- Validation warnings – move your mouse over the individual icons to see the warnings related to the current item.
- Edit Profile Cards – edit the profile cards that are associated to the item. This is mostly relevant if you are a marketing user.

Send feedback about the documentation to docsite@sitecore.net.

The Experience Editor

The Experience Editor is a WYSIWYG editor that allows you to easily make changes to items directly on the page. You can edit all the items that are visible on the page — text, graphics, logos, links, and so on.

In the Experience Editor, you can edit not only the fields of the current item, but also the fields of any items rendered on the page. For instance, if the menu on the page consists of titles of the product items, you can edit the titles without navigating to the product items themselves.

cms_xe_HomePage

The ribbon

In the Experience Editor, the ribbon with all the available functionality is displayed at the top of the webpage. The functionality in the ribbon vary depending on your security roles and the customizations of the website that you are accessing.

If you have full access to all functionality in the Experience Editor, you have access to all five tabs in the ribbon, each with its own collection of commands.

cms_xe_FullRibbon

As a minimum, you can expect to be able to insert a page and delete a page on your website. Regardless of the ribbon that you use, there is always a Save button in the top-left corner.

When you edit an item, you can click the Toggle Ribbon button to hide the ribbon temporarily and give yourself some more space to work with. Click the Toggle Ribbon button again to display the ribbon.

Improve the ribbon load time

You can improve the time it takes to load the Experience Editor ribbon if you turn off the request that displays the number of items you have locked.

Picture 5

To improve the ribbon load time:

- In the `Website\App_Config\Sitecore\Experience Editor\` folder, open the `Sitecore.ExperienceEditor.config` file. In the `WebEdit.ShowNumberOfLockedItemsOnButton` setting, change the value to *false*.

Send feedback about the documentation to docsite@sitecore.net.

The Media Library

You can manage all your media items from the Media Library, such as images that you want to embed in a webpage or make available for download by visitors. The Media Library contains all your media items, such as images, documents, videos, and audio files.

In the Media Library you can:

- Keep all your media files in one place and organize them in a folder structure similar to the content tree.
- Navigate the content tree or use Sitecore's search functionality to quickly find media items.
- Upload media files to the Media Library. You can use the upload buttons to browse your computer for images or simply use the Drag & Drop dialog box to add media files.
- Make every media content item and media file versionable.

Note

To edit your images, you can use Sitecore's Image Editor.

This topic outlines how to:

- Open the Media Library
- Navigate the Media Library user interface

Open the Media Library

You can open the Media Library from the following places.

The Launchpad

To navigate to the Media Library, on the Sitecore Launchpad, click the Media Library icon.

cms_LaunchPad_MediaLibrary

The Content Editor

In the Content Editor, navigate to the Media Library in the content tree. If you want to open the Media Library separately, click Media Library in the page bar at the bottom of the Content Editor.

cms_ce_PageBar_MediaLibrary

The Sitecore Desktop

To open the Media Library, click the Sitecore Start button and then click Media Library.

cms_SitecoreStartMenu_MediaLibrary

Navigate the Media Library user interface

The Media Library user interface consists of:

- The ribbon
- The content tree
- The content area

The ribbon

In the Media Library, you get the same set of options on the ribbon as you do in the Content Editor, although the Insert group allows you to create new media folders and upload media files.

If you select a media item in the content tree, the Media tab displays in the ribbon. This contains the new Media group with the actions that you can perform on the media item.

cms_MediaLibrary_MediaTab

Note

The functionality in the ribbon depends on the user rights you have been assigned by the website administrator and the type of media item you have selected.

The content tree

Use the content tree or the search to find media items. Expand folders in the content tree to see all subfolders and media items and collapse the folders again to hide subfolders.

Make sure you store all your media items in folders and give the folders meaningful names. An easy way to organize your Media Library folder structure is to mirror the structure of your website. For example, if your site has four main sections: *Home*, *Products*, *Jobs,* and *Contact Us* you could create four folders with the same names to store all your images and other media. This will make it easier to locate media items as your site grows and you need to update media items.

The content area

The content area displays the following tabs that you can use when working with a specific media item.

The Search tab

This tab offers an elaborate search functionality that can help you find very specific items among maybe thousands of items in the Media Library. The Media Library opens by default on the Search tab, with a search already run for all items in the Media Library.

cms_MediaLibrary

The Folder tab

This tab displays the options and media contents for the selected folder in the Media Library.

cms_MediaLibrary_ContentArea_Foldertab

The Content tab

This tab displays all the fields that you can edit for the media item that you have selected.

cms_MediaLibrary_ContentArea_ContentTab

Clone and unclone an item

A clone is an item that is not just a copy of the original item, but one that inherits the field values from the original item. If you update a field in the original item, the corresponding field in the clone is also updated.

In Sitecore, you can create clones of items and of entire branches of items. A branch is a parent item and all its descendants.

Clone an item

To create a clone of an item:

1. In the content tree, select the item that you want to clone.
2. On the Configure tab, in the Clones group, click Clone.

   Picture 2

3. In the Clone Item dialog box, select the location where you want to store the clone. You can navigate to the location in the content tree on the Browse tab or search for the location on the Search tab.

Picture 1

4. Click Clone.

If you select the clone item in the Content Editor, you notice that the field titles have the [original value] text added. This means that these values match the corresponding values in the original item and are updated if you edit the field in the original item.

Picture 3

Important

If you clone an item that is in a workflow, the workflow and the Workflow state properties are copied to the clone item. After the workflow fields are copied, the workflow properties of the clone and the workflow properties of the original items are not connected any more. If you change the workflow state of the original item, the workflow state of the clone does not change.

Unclone an item

If you no longer want to have a connection between the original item and the clone, you can unclone the clone item.

To unclone a clone item:

1. Select the clone you want to unclone.
2. On the Configure tab, in the Clones group, click Unclone.

Picture 4

No field values in the clone item change. The connection between the clone and the original item is removed and the clone becomes a normal content item.

Send feedback about the documentation to docsite@sitecore.net.

Clones

In Sitecore, a clone is an item that is not just a copy of the original item, but one that inherits the field values from the original item. If you update a field in the original item, the corresponding field in the clone is also updated.

Therefore, you can think of cloning items as a way of centralizing content items. With cloning, you can place your items throughout the content tree, and control the content from one central location in the content tree. For example, if you have multiple websites in Sitecore – each for a different country, you can create a master website that contains the content items that all the sites have in common. Then, for each of the country websites, you create a clone of the master website that establishes a connection between the original item and the clone, which means that when you update an item in the master website, the corresponding items in all the country websites are also updated.

If you edit a field in a clone and then edit the same field in the original item, a conflict occurs and a notification appears, suggesting how you can resolve the conflict.

Note

Sitecore does not support creating a new clone of a clone item.

Examples

The following examples, describe the connection between the original item and the clones:

- If you edit a field in the original item and save it, Sitecore updates the field in all the cloned items if this field has not been changed in the cloned items.

Original item	Clone
cms_OriginalItem_TitleField	cms_CloneItem_TitleField

Note

In the cloned item, the *[original value]* text next to the value fields indicates that the value has not been changed from the original item.

- If you edit a field in the clone and save it, the *[original value]* text is no longer present next to the Title field of the clone. The original item does not change.
- If you edit a field in the original item and the field has been changed in the clone, a message appears on the Content tab of the clone, telling you that a field in the original item has changed.

cms_CloneItem_FieldHasChangedNotification

To resolve this conflict, you can choose:

- Review the original item

  Displays the original item and you can review the changes.

- Accept the change

  Copies the values from the fields in the original item to the corresponding fields in the clone. The [original value] text appears next to the field titles.

- Reject the change

  Keeps the existing field value in the clone and hides the message.

- If you create a subitem under the original item, a notification appears on Content tab of the clone, telling you that a new subitem has been created under the original item. In the clone, a notification appears telling you to decide whether to create a clone of the new subitem.

cms_CloneItem_NewSubItemNotification

- If you delete an item that has clones, the following message is displayed:

  cms_DeleteItemWIthClones_Notification

  In the message, Sitecore lists all the clones of the item that you are deleting and, if any, the clones of the subitems. If you click OK to delete the original item, the clones keep their current field values and become normal content items.

- If you clone an item that is in a workflow, the workflow and the workflow state properties are copied to the clone. After the workflow field values are copied, the workflow properties of the clone and the workflow properties of the original items are not connected any more. So, if you change the workflow state of the original item, the workflow state of the clone is not changed.

Path to the original item

On the clone in the Content Editor, you can see the path to the original item of the clone. Click the clone in the content tree and on the Content tab, expand the Advanced section, and view the __Source field.

Note

If the Advanced section is not visible, click the View tab and select the Standard fields check box.

cms_CloneItem_SourceField

Create a version of an item

In the Content Editor, you can create new versions of any item in Sitecore, and you can create as many versions of the item as you like.

There are two types of versions:

- Numbered versions – versions of an item in the same language.

  For example, you can create a second version of your products page that you can use during the period of a special promotion. You can also create a new version of an image and change the new version to fit the promotion and then change it back to the original version after the promotion.

- Language versions – versions of an item in a different language.

  For example, you can create a page in Danish and then create an English version of the same page. In this case, you also want to create an additional version of any image if you want to display the Alt text in both languages.

This topic outlines how to:

- Create a numbered version of an item
- Create a language version of an item
- Remove a version of an item

Create a numbered version of an item

To create a new numbered version of an item:

1. In the Content Editor, in the content tree, navigate to the item that you want to add a new version to.
2. Click the Versions tab and in the Versions group, click Add.

   cms_ce_VersionsTab_VersionGroup_AddButton

   Now in the top right corner of the editing pane of the item, you can see the item's new version number. In this example, 2.

cms_ce_ItemVersionNumberButton

3. Make the relevant changes to the new version and save your changes.

Create a language version of an item

To create a new language version of an item:

1. In the Content Editor, in the content tree, navigate to the item that you want to add a language version to.
2. On the Versions tab, in the Language group, click Change the current language and in the drop-down list, select the language that you want the new version to be in.

   Note

   If the language that you want is not in the drop-down list, click More languages, and in the Select Language dialog box, click the relevant language and click OK.

3. In the notification that appears at the top of the item's editing pane, click the link Add a new version.

   cms_ce_AddNewVersionNotification

4. Save your changes.

Remove a version of an item

To remove a version of an item:

1. In the content tree, navigate to the item and the version that you want to remove. Make sure that you select the relevant language and numbered version.
2. In the ribbon, click the Versions tab, and in the Versions group, click Remove, to remove the selected item version.
   - To remove all the versions of the item, click the Remove drop-down arrow, and then click Remove all versions. This does not remove the item itself. Instead, a notification appears on the item, to let you know that there is no version of the item available.

     CE_ContentArea_AddNewVersionNotification

     Send feedback about the documentation to docsite@sitecore.net.

Identify the template that an item is based on

You need to know which template an item is based on, for example, when you want to set up a template as bucketable, or set up RSS feeds.

This topic outlines how to:

- Identify the template that an item is based on
- Enable the Quick Info section

Identify the template that an item is based on

In the Content Editor, in the Quick Info section, you can quickly identify the template that an item is based on.

To identify the template that an item is based on:

1. In the content tree, click the relevant item.
2. In the content area, on the Content tab, expand the Quick Info section at the top. The Template field tells you which template the item is based on.

cms_ContentTab_QuickInfoSection_TemplateField

3. To open the template item, click the template link.

Enable the Quick Info section

If the Quick Info section is not visible, you can enable the section from the Application Options.

1. In the Content Editor, click the System menu , and then click Application Options.

cms_ce_SystemMenu _ApplicationOptions

2. In the Application Options dialog box, in the Show/Hide section, select the Quick Info Section check box and click OK.

cms_ce_ApplicationOptions_ContentEditorTab

Send feedback about the documentation to docsite@sitecore.net.

Searching

The search engine in Sitecore lets you quickly search through all your items in the content tree. To help you locate exactly the item you are looking for, you can use the advanced search functionality, such as search filters and facets.

Note

The search does not include your archived items.

You can access the Sitecore search engine from every item in the content tree in the Content Editor or from the ribbon in the Experience Editor.

The Content Editor

In the Content Editor, there is a Search tab next to the content tab of every item. Item buckets open directly on the Search tab because you need the search functionality to find and open hidden items in an item bucket.

To run a search on an item in the content tree, in the content area, click Search and, in the new Search tab, enter your search criteria. The search runs through the selected item only.

cms_ce_ContentArea_SearchTab

If you want to run more than one search from the same item, you just open a new Search tab. You can open as many search tabs as you want. Each Search tab has a time stamp that gives you a reference to each specific search, for example, if you want to save a search.

The Experience Editor

To open the search in the Experience Editor, in the ribbon, click Search . The Navigate to Item dialog box opens and runs by default a search for all the items in Sitecore.

cms_xe_NavigateToItem_Dialog

The search interface

Whether you access a search from the Content Editor or the Experience Editor, you have the same options to build a query, to view, and to filter the search results.

cms_all_SearchInterface

Search field

In the Search field, you enter the search query for your search. You can enter either free text, use the predefined search filters, or a combination of both. You can access the search options from the drop-down menu.

More search options

If you click the drop-down arrow to the left of the search field, you open a list of pre-defined search options that you can use to optimize and speed up your search. You can, for example, open an item from a list of the most recently modified or opened items, select one of the pre-defined search filters, or apply one of the search operations to all the items returned by the search.

Note

The administrator sets up the search engine and determines, for example, which search views and facets are available to you and whether or not you are able to see media items in the search results.

View types

The different views of the search results are available to the right of the search field. The default views are *List View*, *Grid View*, and *Image View* . If necessary, your administrator can activate several other views such as Tag View or Table View.

Note

If you change the view, the search is launched again, based only on the keywords and filters in the Search field.

Facets

A facet is a way of categorizing the items returned by a search. For example, the items can be classified by author, language, or by when they were created. Once you have a list of search results, you can use the facets to filter the results further.

Search results

When you run a search, the results are listed below the search field. The search results contain basic information about the items returned, such as title, template, and version to give you a quick overview. From the list of search results, you can click an item to open it in the Content Editor or to navigate to the appropriate page in the Experience Editor.

Send feedback about the documentation to docsite@sitecore.net.

Clone and unclone an item

A clone is an item that is not just a copy of the original item, but one that inherits the field values from the original item. If you update a field in the original item, the corresponding field in the clone is also updated.

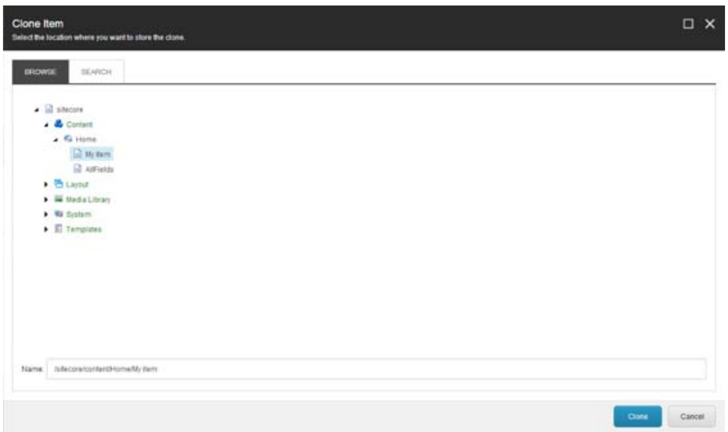
In Sitecore, you can create clones of items and of entire branches of items. A branch is a parent item and all its descendants.

Clone an item

To create a clone of an item:

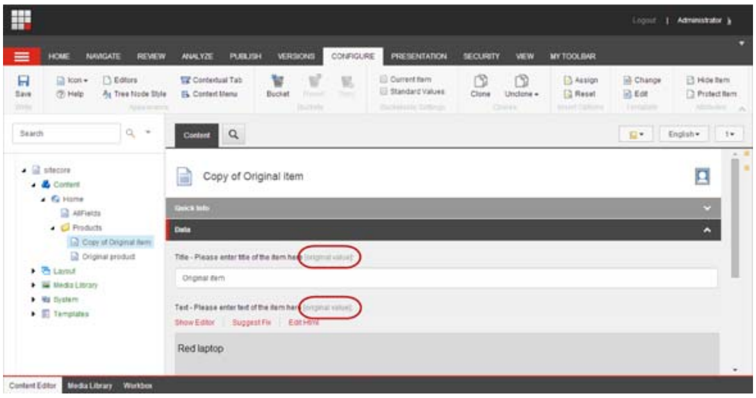
1. In the content tree, select the item that you want to clone.
2. On the Configure tab, in the Clones group, click Clone.

3. In the Clone Item dialog box, select the location where you want to store the clone. You can navigate to the location in the content tree on the Browse tab or search for the location on the Search tab.



4. Click Clone.

If you select the clone item in the Content Editor, you notice that the field titles have the [original value] text added. This means that these values match the corresponding values in the original item and are updated if you edit the field in the original item.



Important

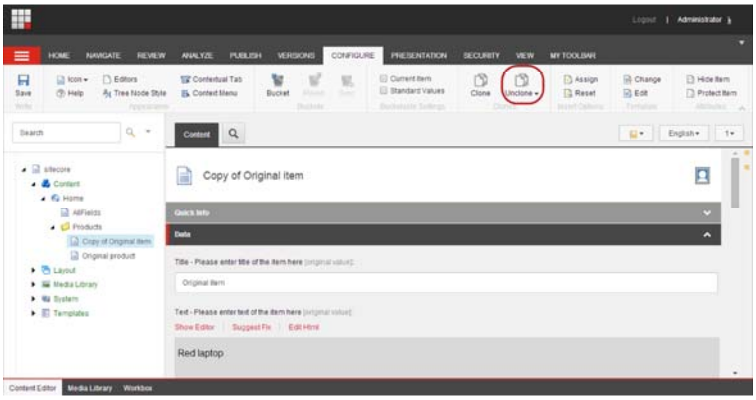
If you clone an item that is in a workflow, the workflow and the Workflow state properties are copied to the clone item. After the workflow fields are copied, the workflow properties of the clone and the workflow properties of the original items are not connected any more. If you change the workflow state of the original item, the workflow state of the clone does not change.

Unclone an item

If you no longer want to have a connection between the original item and the clone, you can unclone the clone item.

To unclone a clone item:

1. Select the clone you want to unclone.
2. On the Configure tab, in the Clones group, click Unclone.



No field values in the clone item change. The connection between the clone and the original item is removed and the clone becomes a normal content item.

Send feedback about the documentation to docsite@sitecore.net.

Clone multiple items

To clone multiple items at the same time, you must use the Sitecore search functionality. In the search field, you can specify the appropriate search criteria or use the facets to find all the items that you want to clone.

Important

When you clone multiple items at the same time, all the clones are placed in the same location.

To clone multiple items:

1. Click the item in the content tree from where you want to start the search for the items that you want to clone.

   Note

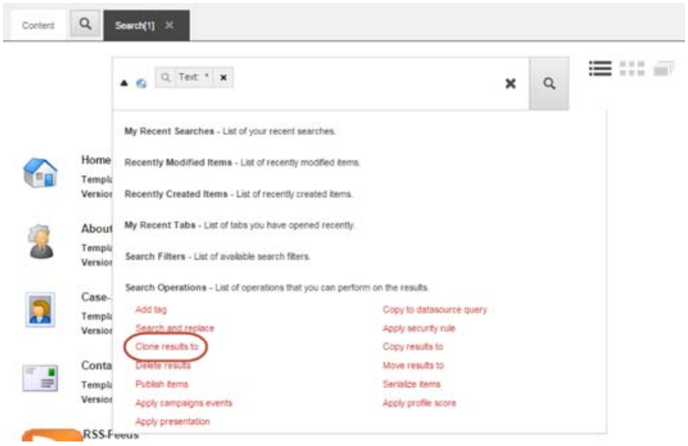
   When you search in Sitecore, the search starts from the selected item in the content tree, and the search results shows up in the order they are found. In this way, the search is quicker and you get most relevant results up first.

2. In the content area, click the Search tab, enter your search keywords, and press ENTER.

   Note

   Use the facets or filters to narrow down your search to list only the items you want to clone.

3. Click the drop-down arrow to the left of the search field, click Search Operations and then click Clone results to.



4. In the Clone Item dialog box, select the location in the content tree where you want to place the clones. On the Browse tab, you can navigate to the location in the content tree, or you can search for the location on the Search tab.
5. Click Clone.

Send feedback about the documentation to docsite@sitecore.net.

Clones

In Sitecore, a clone is an item that is not just a copy of the original item, but one that inherits the field values from the original item. If you update a field in the original item, the corresponding field in the clone is also updated.

Therefore, you can think of cloning items as a way of centralizing content items. With cloning, you can place your items throughout the content tree, and control the content from one central location in the content tree. For example, if you have multiple websites in Sitecore – each for a different country, you can create a master website that contains the content items that all the sites have in common. Then, for each of the country websites, you create a clone of the master website that establishes a connection between the original item and the clone, which means that when you update an item in the master website, the corresponding items in all the country websites are also updated.

If you edit a field in a clone and then edit the same field in the original item, a conflict occurs and a notification appears, suggesting how you can resolve the conflict.

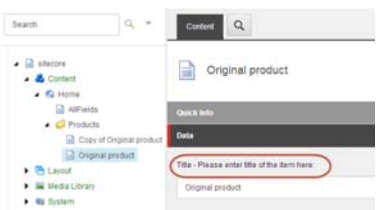
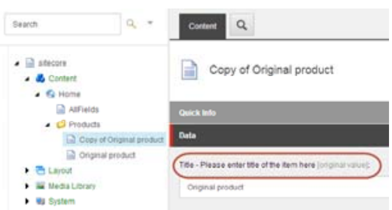
Note

Sitecore does not support creating a new clone of a clone item.

Examples

The following examples, describe the connection between the original item and the clones:

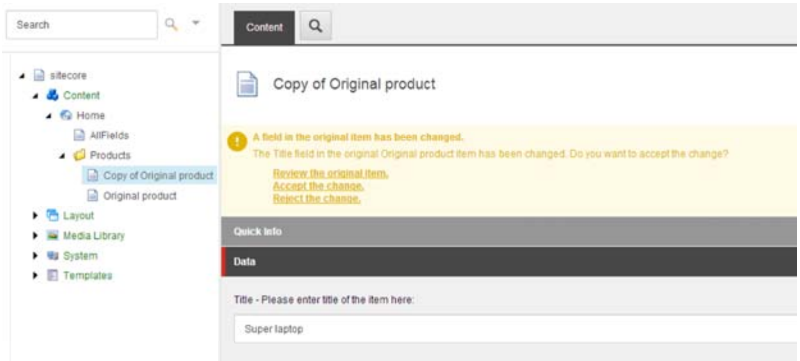
- If you edit a field in the original item and save it, Sitecore updates the field in all the cloned items if this field has not been changed in the cloned items.

Original item	**Clone**
	

Note

In the cloned item, the *[original value]* text next to the value fields indicates that the value has not been changed from the original item.

- If you edit a field in the clone and save it, the *[original value]* text is no longer present next to the Title field of the clone. The original item does not change.
- If you edit a field in the original item and the field has been changed in the clone, a message appears on the Content tab of the clone, telling you that a field in the original item has changed.



To resolve this conflict, you can choose:

- Review the original item

  Displays the original item and you can review the changes.

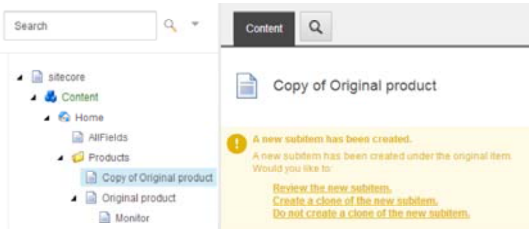
- Accept the change

  Copies the values from the fields in the original item to the corresponding fields in the clone. The [original value] text appears next to the field titles.

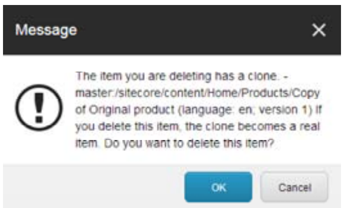
- Reject the change

  Keeps the existing field value in the clone and hides the message.

- If you create a subitem under the original item, a notification appears on Content tab of the clone, telling you that a new subitem has been created under the original item. In the clone, a notification appears telling you to decide whether to create a clone of the new subitem.



- If you delete an item that has clones, the following message is displayed:



  In the message, Sitecore lists all the clones of the item that you are deleting and, if any, the clones of the subitems. If you click OK to delete the original item, the clones keep their current field values and become normal content items.

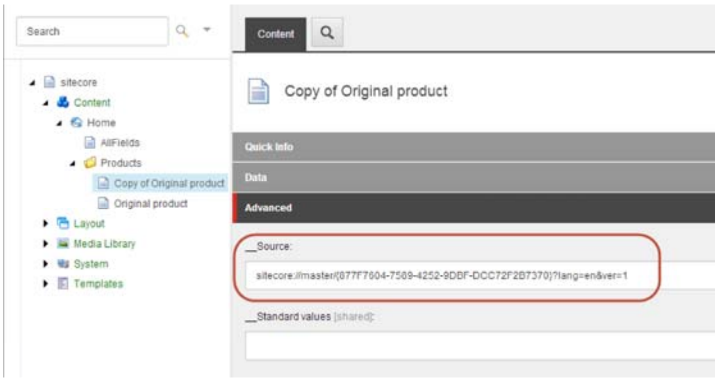
- If you clone an item that is in a workflow, the workflow and the workflow state properties are copied to the clone. After the workflow field values are copied, the workflow properties of the clone and the workflow properties of the original items are not connected any more. So, if you change the workflow state of the original item, the workflow state of the clone is not changed.

Path to the original item

On the clone in the Content Editor, you can see the path to the original item of the clone. Click the clone in the content tree and on the Content tab, expand the Advanced section, and view the __Source field.

Note

If the Advanced section is not visible, click the View tab and select the Standard fields check box.



Send feedback about the documentation to docsite@sitecore.net.

The impact of versions on clones

If you want to clone an item that has multiple versions or if you want to have more than one version of a clone, you must be aware of the specific rules that define how the clones are updated.

Action	Effect
Cloning an item that has multiple versions	The clone links to the version of the original item that you selected when you created the clone. Before cloning the item, make sure to select the appropriate version of the original item.
Adding a new version to the original item	A notification appears on the clone, asking you to decide whether you want to create a corresponding new version of the clone.
Adding a new version of the clone	The new version links to the version of the original item that the previous version of the clone linked to.
Updating a version of the original item	Only the clone versions that are linked to that specific version of the original item are updated accordingly.
Removing a version of the original item that has clones	All the clone versions that are linked to that specific version of the original item are uncloned. This means that these versions of the clone are turned into regular items that no longer inherit any values from the original item.

Send feedback about the documentation to docsite@sitecore.net.

View and compare item versions

If there is more than one version or more than one language version of an item, you can get an overview of all the versions available in the Content Editor. In addition, if you are unsure about what the differences are between two versions of the same item, you can compare the versions field by field.

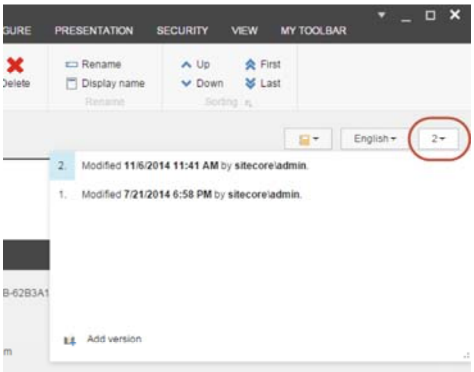
This topic outlines how to:

- View the versions of an item
- Compare the versions of an item

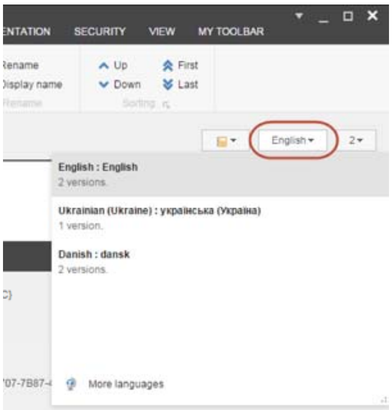
View the versions of an item

To view the versions or the language versions of an item:

1. In the Content Editor, navigate to the item that you want to view the versions for.
    - To view the available versions of the item, click Version number 2▾ in the upper right corner of the content area.



- To view the language versions of the item, click Language version in the upper right corner of the content area.



2. To open another version, click the relevant version in the Version number or Language version drop-down menu.

   Note

   In the Version number drop-down menu, only the versions available in the selected language appear.

Compare the versions of an item

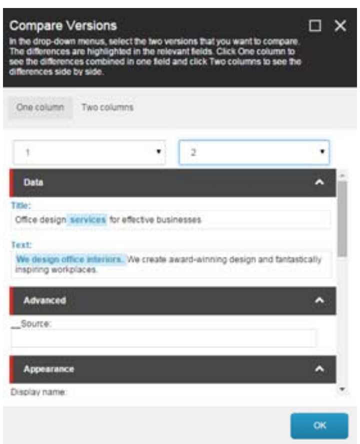
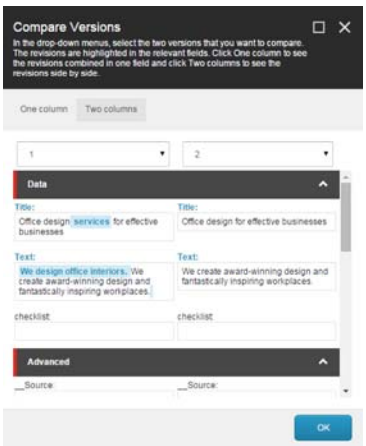
To compare two different versions of an item:

1. In the Content Editor, navigate to the item that you want to compare the versions for.

   Important

   If the item has multiple language versions, make sure that you also select the relevant language for the item.

2. In the ribbon, click the Versions tab, and in the Versions group, click Compare.
3. In the Compare Versions dialog box, in the drop-down menus, select the version numbers that you want to compare. You can compare two versions at a time.
4. Click One column to see the differences highlighted in the same field or click Two columns to see them side by side.

One column	Two column
	

Note

The sections and fields in the Compare Versions dialog are the same as you see in the content area for the item. You can collapse or expand the sections as needed.

Send feedback about the documentation to docsite@sitecore.net.

Add a tag to an item

Tagging your items is a great way to categorize them and make it easy to search and find items that have the same tag. In particular, when you have potentially millions of items in an item bucket, using tags can be the quickest way to find your items.

Note

To search for items with a specific tag, you can use the search filter Search by Tag in the Sitecore search functionality.

This topic outlines how to:

- Add a tag to a single item in the Content Editor
- Add a tag to multiple items at the same time using the search operation

Add a tag to a single item in the Content Editor

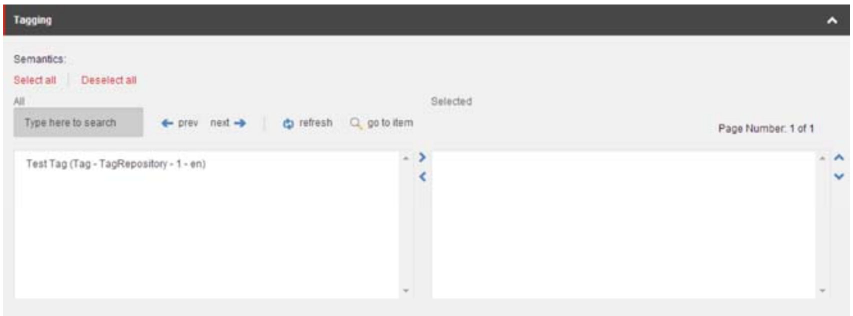
To add a tag to an item:

1. In the Content Editor, in the content tree, navigate to the item that you want to add a tag to.
2. In the content area, on the Content tab, find and expand the Tagging section.

   Note

   If you do not see the Tagging section, click the View tab and, in the View group, select the Standard fields check box.

3. In the Tagging section, in the list of all the available tags on the left, you can either search for a specific tag in the search field or select a tag manually. You can use SHIFT or CTRL to select multiple tags.

   Note:

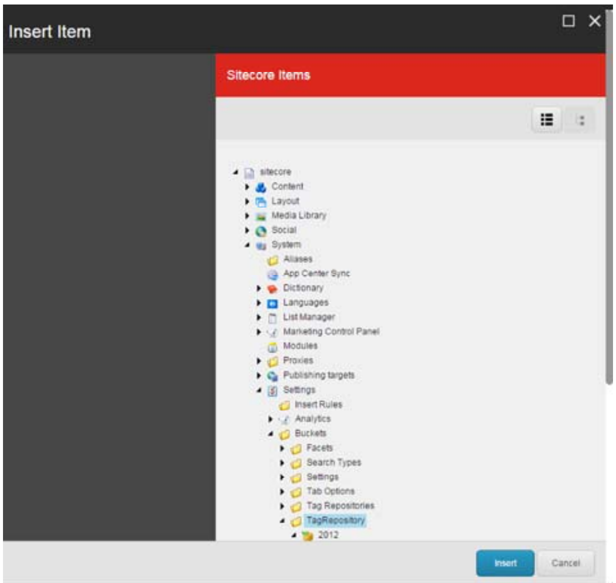
   Your Sitecore administrator specifies the tags that are available to you.

4. When you have selected the relevant tags for the item, click the right arrow to move them to the list of selected tags.
5. To remove tags from the Selected list, click the left arrow or double-click a tag to move it from one field to the other.
6. Save your changes.

Add a tag to multiple items using the search operation

To add a tag to multiple items at the same time using the search operation:

1. In the Content Editor or the Experience Editor, create a search for the items that you want to add a tag to.
2. When all the relevant items are listed in your search results, click the drop-down arrow ▼ to the left of the Search field.
3. In the drop-down menu, click Search Operations and then click Add Tag.
4. In the Insert Sitecore Item dialog box, in the content tree, navigate to the tag that you want to add to all your search results or click List view to search for the tag.



5. To add the tag to all the items listed in your search, click Insert.

Send feedback about the documentation to docsite@sitecore.net.

Archive and restore an item

You can archive items that you do not want to display on your website in the Content Editor. When you archive an item, it is not deleted but stored outside the content tree. Archived items can be restored at any time.

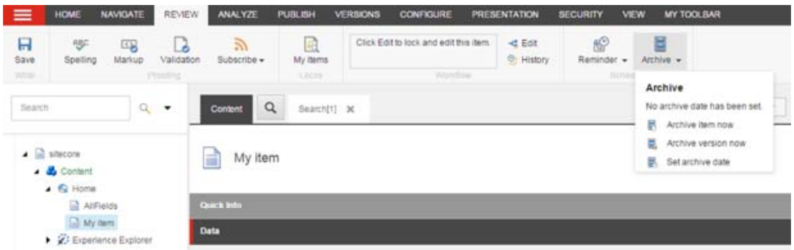
This topic outlines how to:

- Archive an item immediately
- Archive an item at a later date
- Restore an archived item

Archive an item immediately

To archive an item:

1. In the Content Editor, in the content tree, navigate to the item that you want to archive.
   - If you want to archive a version of an item, click the Versions button and select the relevant version.
   - If you want to archive an item in a specific language, click the Language button and select the relevant language.
2. To archive the item immediately, on the Review tab, in the Schedule group, click Archive:
   - To archive the item in all versions and languages, including all its subitems, click Archive item now.
   - To archive the current version and language of the item, click Archive version now.



3. Save your changes.

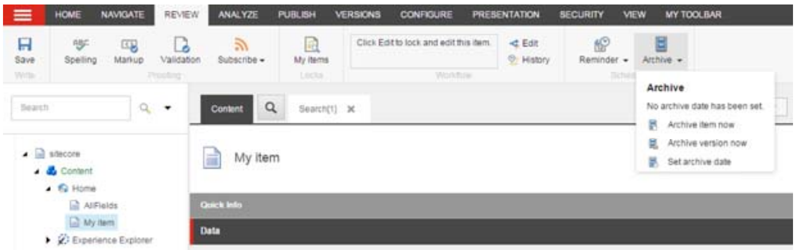
   Note

   When you archive an item, it is removed from the content tree, but it remains on the published website. Therefore, you must republish your website to ensure that your changes are replicated on your website.

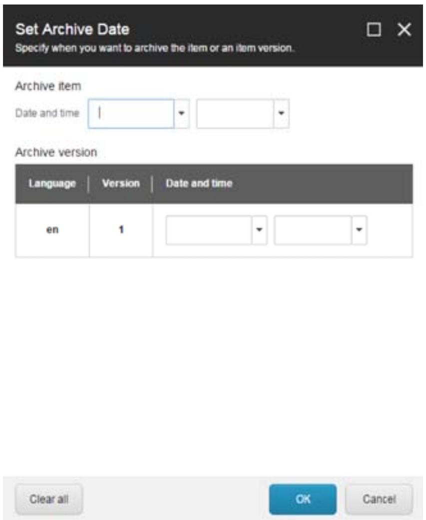
Archive an item at a later date

To archive an item:

1. In the Content Editor, in the content tree, navigate to the item that you want to archive.
   - If you want to archive a version of an item, click the Versions button and select the relevant version.
   - If you want to archive an item in a specific language, click the Language button and select the relevant language.
2. To schedule the archive of the item on a different date, on the Review tab, in the Schedule group, click Archive and then click Set archive date.



3. In the Archive Date and Time dialog box, specify the date and time you want to archive the item.
    - In the Archive item section, specify the date and time that you want to archive the item including all its versions and subitems.
    - In the Archive version section, specify when you want to archive a specific version of the current item.



When you set a date and time, it is important to remember that:

    - If you only specify a date, the item is archived at 12:00 midnight on that date.
    - If you only specify a time, the item is archived at that time on the current date.
    - If you specify a date or a time that is in the past, the item is archived the next time that Sitecore checks for items to archive.
4. Click OK and save your changes.

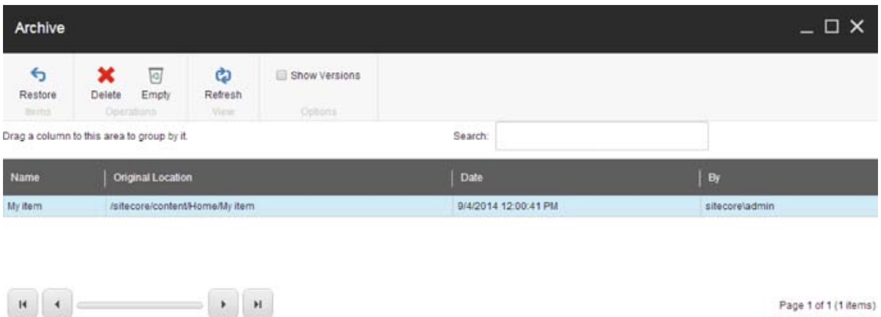
    Note

    When you archive an item, it is removed from the content tree, but it remains on the published website. Therefore, you must republish your website to ensure that these changes are replicated on your website.

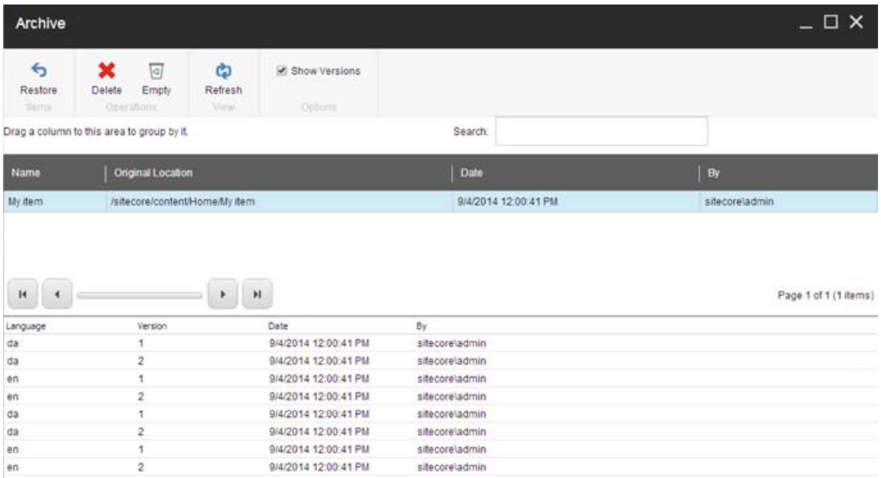
Restore an archived item

To restore a content item from the archive:

1. In the Sitecore Desktop, click the Sitecore Start button, All Applications, and then click Archive.
2. In the Archive dialog box, select the item that you want to restore.
    - To restore all versions and languages of an item, including the subitems, select the item from the list. You can only restore one item at a time.



    - To restore a version of an item, in the Options group, select Show Versions, click the item that contains the version that you want to restore and in the lower pane that appears, select one or more versions to restore.



Note

You can select and restore as many versions of the same item as you want.

3. In the Items group, click Restore and the items are restored in the content tree.

Send feedback about the documentation to docsite@sitecore.net.

Edit or change the template that an item is based on

You can edit or change the template that an item is based on, for example, if you are using a predefined branch template, and you want to change the template that one of your subitems is based on.

Note

When you edit a template, all the items that are based on the template will be affected by the changes you make.

To edit or change the template that an item is based on:

1. In the Content Editor, navigate to the relevant item.
2. On the Configure tab, in the Template group, click:
   - Change – to change the template. In the Select the Template dialog that opens, navigate to or search for the template that you want to change to and follow the steps in the wizard.
   - Edit – to edit the template. In the Template Manager that opens, click the Builder tab and make the relevant changes to the data template fields.
3. Save your changes.

Send feedback about the documentation to docsite@sitecore.net.

Identify the template that an item is based on

You need to know which template an item is based on, for example, when you want to set up a template as bucketable, or set up RSS feeds.

This topic outlines how to:

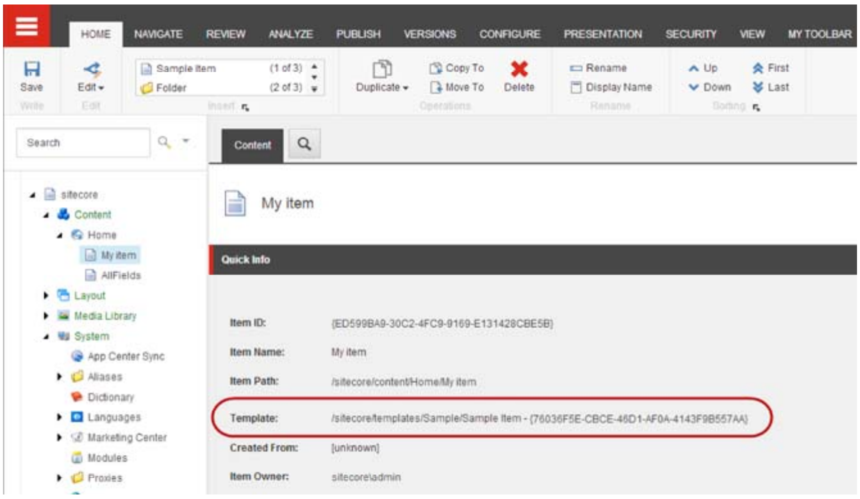
- Identify the template that an item is based on
- Enable the Quick Info section

Identify the template that an item is based on

In the Content Editor, in the Quick Info section, you can quickly identify the template that an item is based on.

To identify the template that an item is based on:

1. In the content tree, click the relevant item.
2. In the content area, on the Content tab, expand the Quick Info section at the top. The Template field tells you which template the item is based on.

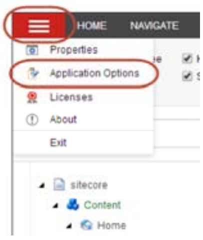


3. To open the template item, click the template link.

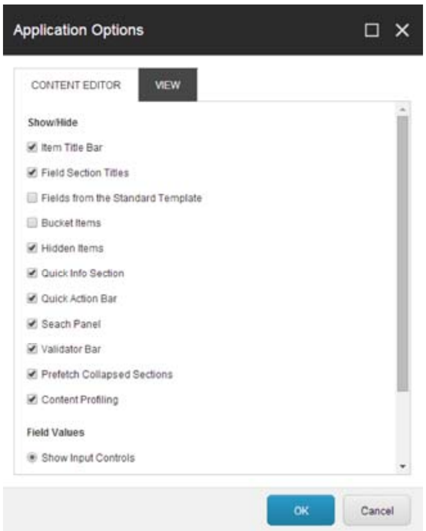
Enable the Quick Info section

If the Quick Info section is not visible, you can enable the section from the Application Options.

1. In the Content Editor, click the System menu, and then click Application Options.



2. In the Application Options dialog box, in the Show/Hide section, select the Quick Info Section check box and click OK.



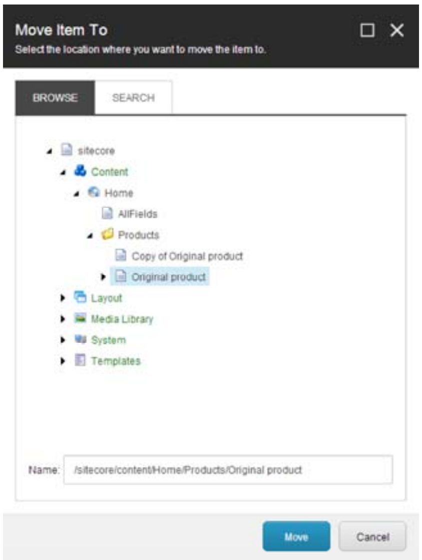
Send feedback about the documentation to docsite@sitecore.net.

Move an item

You can move an item to a different location in either the Content Editor or the Experience Editor.

To move an item:

1. In the Content Editor or the Experience Editor, navigate to the item or page that you want to move.
   - In the Content Editor, on the Home tab, in the Operations group, click Move to. Alternatively, you can right click the item in the content tree, click Copying and then click Move to.
   - In the Experience Editor, on the Home tab, in the Edit group, click Move page.
2. In the Move Item To dialog box, select the location that you want to move the item to. You can navigate to the location in the content tree on the Browse tab or search for the location on the Search tab.



3. Click Move.

Send feedback about the documentation to docsite@sitecore.net.

Restore or remove a deleted item

When you delete an item, the item is moved to the Sitecore Recycle Bin from where you can:

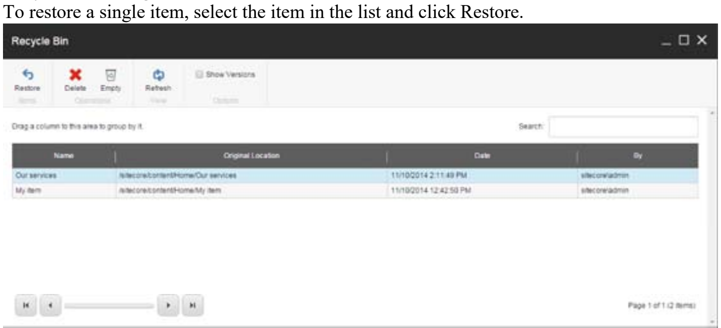
- Restore an item or a version of the item.
- Permanently delete individual items.
- Empty the entire bin and permanently delete all items.

To restore or remove an item from the Sitecore Recycle Bin:

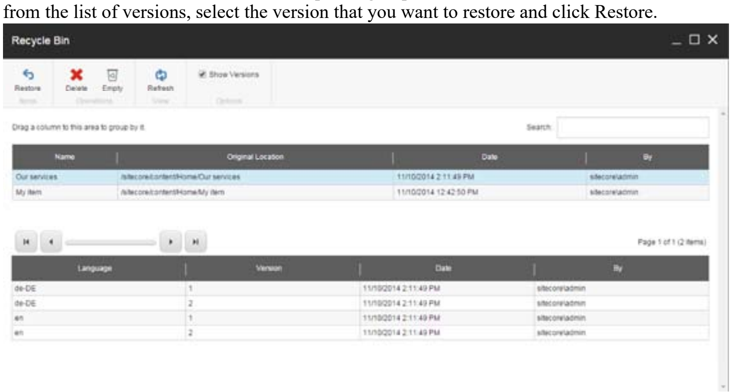
1. Open the Sitecore Launch Pad and click Recycle Bin.

   Alternatively, you can open the Recycle Bin from the Sitecore Start menu.

2. In the Recycle Bin dialog box:
   - To restore a single item, select the item in the list and click Restore.

   - To restore a version of the item, in the Options group, select the Show Versions check box to display all the deleted versions of the selected item. Then from the list of versions, select the version that you want to restore and click Restore.

   Note

   When you restore an item or a version of an item, it is restored to its original place in the content tree.

   - To permanently delete an individual item, select the item that you want to delete and in the Operations group, click Delete.
   - To empty and permanently delete the entire recycle bin, in the Operations group, click Empty.

3. Click OK.

Send feedback about the documentation to docsite@sitecore.net.

Convert an item into an item bucket

Item buckets let you manage large numbers of items in the content tree. An item bucket is a container that you can use to hide innumerable items and still easily retrieve and work with these items. The only way to find bucket items is to use the Sitecore search engine.

You can convert any item into an item bucket, whether it is a new item or an existing item.

When you convert an item into an item bucket, the item bucket organizes and hides all its subitems if they are bucketable items or based on a template that is bucketable. When you convert an item that contains many subitems into an item bucket, it can take a long time to organize the items. A progress bar appears and displays a running tally of the items that are being processed. During bucketing, you can cancel the process before it is complete.

Note

On a standard PC, organizing 1000 items into an item bucket takes approximately 5-10 minutes.

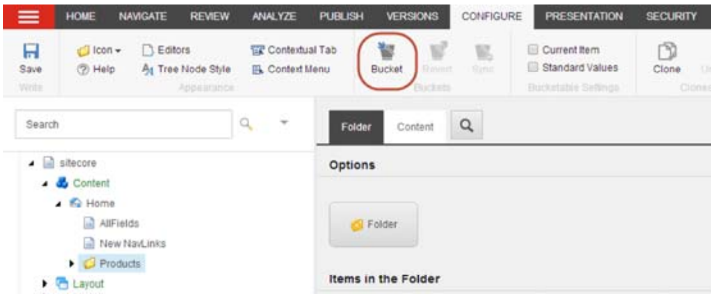
This topic outlines how to:

- Convert an item into an item bucket
- Revert an item bucket into a regular item

Convert an item into an item bucket

To convert an item into an item bucket:

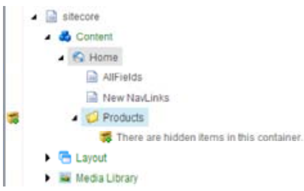
1. In the Content Editor, click the item that you want to convert into an item bucket.
2. On the Home tab, click Edit to lock the item.
3. Click the Configure tab, and then in the Buckets group, click Bucket to convert the item into an item bucket.



Note

If the process of making an item bucketable fails, you can synchronize the item bucket and the process picks up where it left off.

To see which items are item buckets in the Content Editor, you can right-click the Quick Action Bar to the left of the content tree and select Item Buckets to display an icon appears next to each item bucket in the content tree. In addition, if you expand an item bucket in the content tree, you can see a visible cue that the item is an item bucket. If the items in an item bucket are hidden, a small notification tells you that there are hidden items in the container.



Revert an item bucket into a regular item

You can always revert the item bucket back into a regular item – even after you have added items to the item bucket.

To revert an item bucket into a regular item:

1. In the Content Editor, navigate to the item bucket that you want to revert into a regular item.
2. On the Configure tab, in the Buckets group, click Revert. If you have many subitems in the item bucket, this may take some time.

   The item bucket reverts to a regular item, and all its subitems are displayed as regular items and they maintain the parent-child relationship with their ancestors and descendants.

Important

If you delete an item bucket, *all* the items in the bucket, whether they are hidden or visible, are deleted as well. To restore a deleted item bucket and its items, you select the item bucket in the Sitecore Recycle Bin and click Restore.

Send feedback about the documentation to docsite@sitecore.net.

Item buckets

Item buckets let you manage large numbers of items in the content tree. An item bucket is a container that you can use to hide innumerable items and still easily retrieve and work with these items. The only way to find bucket items is to use the Sitecore search engine.

As your content tree grows, it gets more difficult to locate items, and performance can decrease. For example, if you have a website that markets or sells innumerable products or uses an ever-increasing number of media files, as you continually create more items, your content tree becomes unmanageable. If you use an item bucket to store your items instead, you can store an unlimited number of items in one container without displaying them in the content tree.

Using item buckets has many advantages, including:

- All the content items in an item bucket are automatically organized in a logical format that increases the performance of the search engine.
- A single item bucket can contain millions of content items without slowing down the UI or congesting the content tree.
- You can have as many item buckets as you want. This lets you split up your item buckets into logical containers, for example, one for products and one for articles.

This topic contains information about:

- Store items in an item bucket
- Synchronize an item bucket
- Search for hidden items
- View hidden items

Store items in an item bucket

An item bucket can contain both regular items and bucketable items:

- Regular items (structured items) are stored in an item bucket, are visible in the content tree, and they maintain their parent-to-child relationship with their ancestors and descendants. These items are based on a template that does not support item buckets.
- Bucketable items (unstructured items) are hidden in the content tree and do not maintain their parent-to-child relationship. These items are based on a template that supports item buckets.

  Important

  You should not store both regular and bucketable items in an item bucket. This can introduce unnecessary complexities.

When you store items in an item bucket, the parent-to-child relationship between the items is removed, and instead the items are organized in a folder structure according to the date and time they were created. Therefore, the item buckets feature is primarily designed for storing content items that do not need to be stored in a hierarchy.

Note

Your Sitecore developer can configure your Sitecore solution to organize the items in the item bucket differently, for example, by their globally unique identifier (GUID).

To decide whether to turn an item into an item bucket, you must decide whether you need to keep the hierarchy of your items in the content tree. If you do not need subitems of an item to be stored in a particular way and if you have many items, item buckets can be a good idea.

Synchronize an item bucket

Whenever you change the bucketable settings for items in an item bucket, you must synchronize the item bucket to make sure that all the items are hidden or shown according to whether they are bucketable or not.

You must synchronize an item bucket when you:

- Make a regular item that is stored in an item bucket into a bucketable item.
- Make a bucketable item into a normal item.
- Make a template bucketable.
- Change a bucketable template into a normal template.
- Delete an item from the item bucket.

Search for hidden items

When bucketable items are stored in an item bucket, they are hidden in the content tree by default. This means that when you want to work with a bucketable item located in an item bucket, you need to use the Sitecore search functionality to find and open it from the list of search results.

When you convert an item into an item bucket, the item displays the Search tab in the editing pane where you can search for all the items in an item bucket, whether they are hidden bucketable items or regular items. Sitecore offers a wide range of search filters, search operations, and facets to help you find the content item that you are looking for.

cms_ce_ItemBucket_HiddenItems

View hidden items

If you need to view the bucketable items that are hidden in an item bucket, in the Content Editor, click the View tab, and in the View group, select the Buckets check box.


cms_ce_Ribbon_ViewTab_ViewGroup_BucketsCheckBox

However, you should clear the Buckets check box when you work with item buckets. This prevents the system from unnecessarily loading all the items in the content tree and thereby slowing down performance. You can still work with the hidden content items by searching for them on the Search tab, which is available for every item bucket.

Send feedback about the documentation to docsite@sitecore.net.

Make an item a bucketable item

There are two ways to make an item a bucketable item and thereby hide the item if it is placed in an item bucket. You can:

- Make the individual item bucketable.

  You can make only one item bucketable at a time. In this way, you do not make any changes to the template that will influence other items.

- Make the template that the item is based on bucketable.

  When you make the template bucketable, all the items that are based on this template are stored as hidden items if they are placed in an item bucket. For this to take effect, you have to synchronize each item bucket.

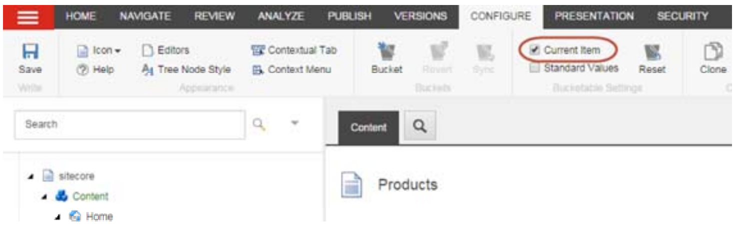
This topic outlines how to:

- Make an individual item bucketable
- Make a template bucketable
- Make a template bucketable using the Template Manager
- Make an item or template no longer bucketable using search
- Make a template no longer bucketable using the Template Manager

Make an individual item bucketable

To make an individual item into a bucketable item:

1. In the Content Editor, select the item that you want to hide in an item bucket.
2. On the Configure tab, in the Bucketable Settings group, select the Current item check box.



3. To make bucketable settings take effect in the item bucket, you must synchronize the item bucket. In the content tree, select the item bucket where the item is stored, and in the Buckets group, click Sync.

   This completes the process and hides the item if it is in an item bucket.

Note

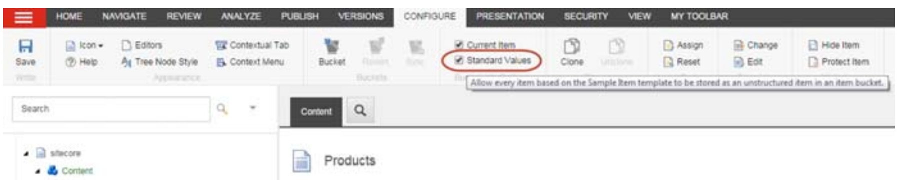
If you want to open a bucketable item that is placed in an item bucket, you must click the item bucket and use the Sitecore search functionality to locate and open the item.

Make a template bucketable

If you have a large number of similar items that you want to make bucketable and therefore hidden if they are in an item bucket, you can make the template that they are based on bucketable.

To make a template bucketable in the Content Editor:

1. In the Content Editor, navigate to one of the items that is based on the template that you want to make into a bucketable template.
2. On the Configure tab, in the Bucketable Settings group, select the Standard Values check box, to make every item that is based on the selected item's template bucketable.



Note

To verify the name of template that you want to make bucketable, you can hover over the Standard Values check box, and the name of the template is in the tooltip that appears.

3. In the content tree, select the item bucket that contains items based on this template, and then in the Buckets group, click Sync.

   All items that are based on this template are now hidden if they are located in the item bucket.

4. Synchronize all the item buckets that contain items based on this template. This updates their structure and hides the bucketable items in the content tree.

Note

If you create items based on this template in a folder that is not an item bucket, these items are treated like normal items and are displayed in the content tree.

Make a template bucketable using the Template Manager

Alternatively, if you know the name of the template that you want to make bucketable, you can open the Template Manager and make the template bucketable from there.

To make a template bucketable in the Template Manager:

1. To open the Template Manager, in Sitecore Desktop, click the Sitecore Start button, and then click Template Manager.
2. In the content tree, navigate to the template that you want to make bucketable. Expand the template and select the _Standard Values item.
3. On the Configure tab, in the Bucketable Settings group, select the Current Item check box.
4. In the content tree, select the item bucket that contains items based on this template, and then in the Buckets group, click Sync.

   All items that are based on this template are now hidden if they are located in the item bucket.

5. Synchronize all the item buckets that contain items based on this template. This hides the bucketable items in the content tree.

Note

If you create items based on this template in a folder that is not an item bucket, these items are treated like normal items and are displayed in the content tree.

Make an item or an item's template no longer bucketable

If you want to make the bucketable item or the template that the item is based on no longer bucketable, you must first use the search functionality to locate the item in the item bucket.

To make a bucketable item or the item's template no longer bucketable:

1. To locate the item, select the item bucket where the item is stored and on the Search tab, in the search field, enter a word that identifies the item.
2. In the search results, click the item to open it.
3. In the ribbon, on the Configure tab, in the Bucketable Settings group:
   - Clear the Current Item check box to make the item no longer bucketable.
   - Clear the Standard Values check box to make the template no longer bucketable.
4. To make bucketable settings take effect in the item bucket, select the item bucket where the item is stored and on the Configure tab, in the Buckets group, click Sync.

When synchronization is complete, the item or all the items that are based on the template are displayed as regular items in the item bucket that you synchronized.

Note

If you made a template no longer bucketable, you must synchronize each item bucket that contains items based on the template that you changed.

You can also change a bucketable template back into a normal template in the Template Manager.

Make a template no longer bucketable using the Template Manager

To make a bucketable template no longer bucketable:

1. To open the Template Manager, in the Sitecore Desktop, click the Sitecore Start button, and then click Template Manager.
2. In the content tree, navigate to the template that you want to change.
3. Expand the template and select the _Standard Values item.
4. On the Configure tab, in the Bucketable Settings group, clear the Current item check box to change the bucketable template to a normal template.
5. In the Content Editor, select an item bucket that contains items based on this template and on the Configure tab, in the Buckets group, click Sync to synchronize the item bucket.

When synchronization is complete, all the items that are based on this template are displayed as regular items in the item bucket that you synchronized.

You must synchronize each item bucket that contains items based on the template that you changed.

Send feedback about the documentation to docsite@sitecore.net.

Save a search query for an item bucket

When you work with item buckets, you might regularly run the same search query from the same item bucket. In the Content Editor, you can save a search on the item bucket, which means that every time you click an item bucket, the search runs the saved search query automatically.

Note

The search queries that you save on an item bucket are unique to your own Sitecore account. However, your Sitecore administrator should set up all the saved searches and determine whether they should be available to a specific user or to all users.

To save a search query on an item bucket:

1. In the content tree, navigate to the item bucket where you want to save a search.
2. On the Home tab, in the Insert group, click Saved Search. Alternatively, you can right-click the item bucket, click Insert and then Saved Search.
3. In the Message dialog box, enter a suitable name for the saved search and click OK. The saved search item is then added to the content tree as a subitem of the item bucket.
4. In the content tree, click the saved search item and on the Content tab scroll down to the Item Buckets section.

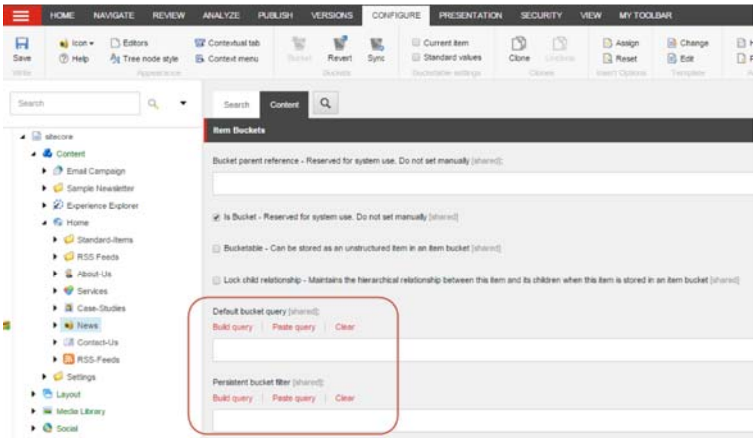
   Note

   If you do not see the Item Buckets section, in the ribbon, click the View tab and in the View group, select the Standard Fields check box.

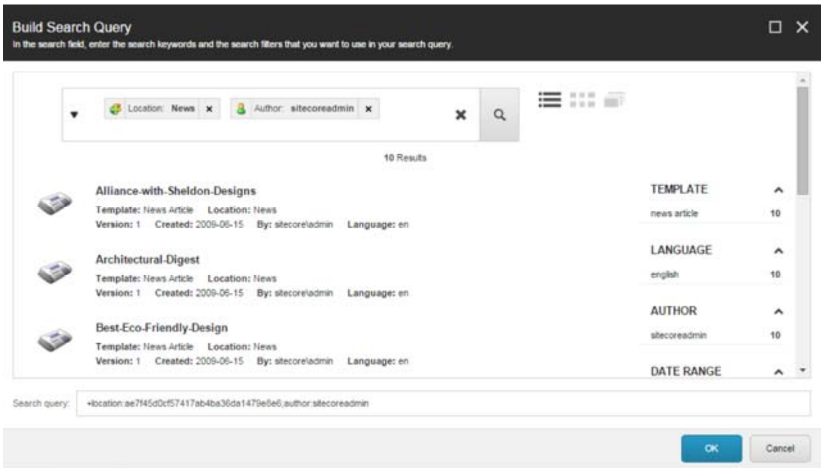
5. Click Build Query above the query you want to build.

   You can build either a:

   - Default Bucket Query – a query that appears in the search field every time you open the search tab. If necessary, you can delete the query from the search field. The default query is 'text:*', which results in returning everything.
   - Persistent Bucket Filter – a query that appears in the search field every time you open it and that you cannot delete. The default query consists of the 'location:{ID of the parent item}', which states the location from where the search should be run, and the '-template:{ID of the saved search template}', which states that items based on the saved search template should not be displayed.

6. In the Build a Search Query dialog box, in the Search field, build your search query by adding the relevant filters and search keywords and then click OK. For example, the search query in the screenshot below consists of the persistent bucket filter for location and author, but you can enter as many filters as you need.

7. To remove any queries from the Default Bucket Query or the Persistent Bucket Filter fields, click Clear above the query field that you want to clear.

8. Save your changes.

Now, when you click the item bucket in the content tree, the search runs automatically.

Send feedback about the documentation to docsite@sitecore.net.

RSS feeds

An RSS (Really Simple Syndication) is a web feed that can publish frequently updated material — such as blogs, news headlines, audio, video, and so on — in a standardized format. An RSS feed can contain either the full text of the material or a summarized version, as well as metadata, such as, publishing dates, and authorship.

In Sitecore, there are two types of RSS feeds:

- RSS feeds – feeds that website visitors can subscribe to.
- Client RSS feeds – feeds in the Content Editor that Sitecore users can subscribe to.

RSS feeds

With RSS feeds, publishers can make a summary or update of recently added content available to the website visitor. In this way, the visitors can subscribe to these feeds and automatically receive updates from their favorite websites.

A visitor subscribes to a feed by clicking an RSS icon in a browser and this initiates the subscription process. The RSS reader checks the visitor's subscribed feeds regularly to see if there are any updates to download, and provides the visitor with an interface to monitor and read the feeds.

Client RSS feeds

In the Content Editor, you can use Client RSS feeds to stay informed about any item that you are interested in. Unlike the regular RSS feeds, you can subscribe to a Client RSS feed for any item in the Content Editor and receive updates on all the changes that are made to the item. In addition, if you have the appropriate permissions, you can run workflow commands directly from the item feed and move the item through the workflow.

Sitecore contains the following Client RSS feeds that content authors and other users of Sitecore can subscribe to:

- Item feeds – keeps you up to date on the entire item and is useful for content authors.
- Workflow feeds – this keeps you up to date on the entire workflow and is useful for administrators and super users who have access to every workflow state that the workflow contains.
- Workflow state feeds – this client feed keeps you up to date on this particular workflow state. This client feed is useful for content authors and managers who only have access to particular workflow states.

Security and Client RSS feeds

The RSS technology allows users who follow a link in an RSS feed to be directed to the specific item.

Most RSS readers do not support authentication, which means that users who subscribe to a Sitecore Client RSS feeds have direct access to the item that is specified in the URL of the RSS feed – they do not have to identify themselves to the Sitecore security system. However, the Sitecore security system verifies that they are authorized users if they try to perform any actions associated with the feed.

If an unauthorized user gains access to the URL of the RSS feed:

- They can follow the link and view all the content contained in the RSS feed even though their own security permissions do not give them access to this item.
- They cannot perform any actions on the content.
- They cannot view any other content.
- They cannot gain access to the username or password of the original owner of the RSS feed.
- They cannot modify the link to gain access to any other content.

Important

Sitecore users should not share RSS feeds.

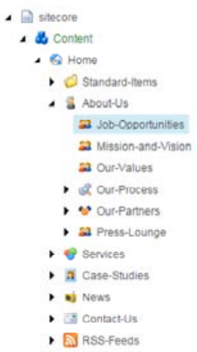
Send feedback about the documentation to docsite@sitecore.net.

Set up RSS feed renderings for subitems

If you create an RSS feed for an item where the subitems are based on different templates, you must specify how each of these subitems are displayed in the feed.

For example, you use your website to promote a range of products. In the content tree, all the different products are stored under the parent item *Products* and they are not all based on the same template.

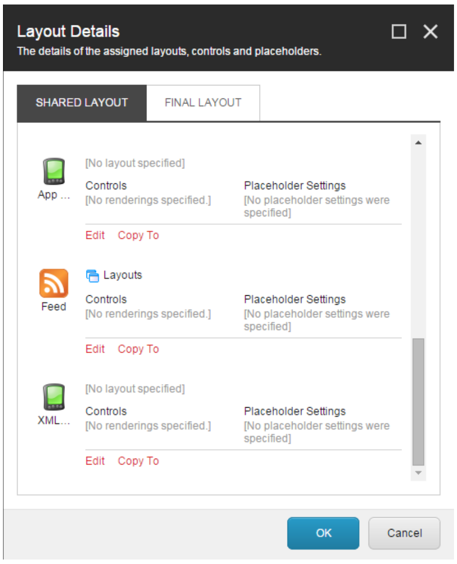
The content tree might look something like this:



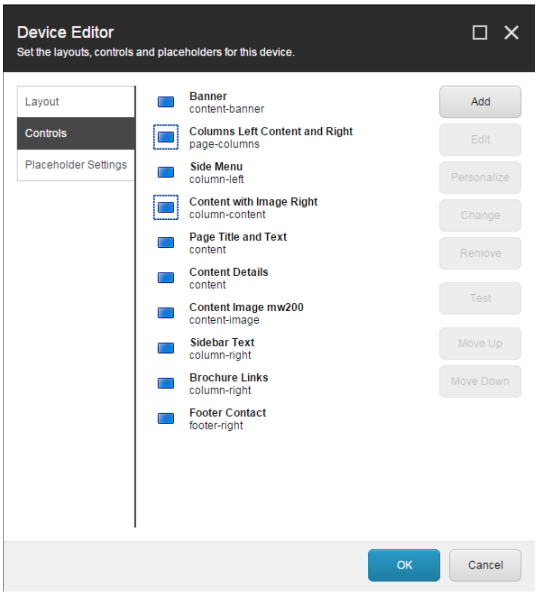
When you create an RSS feed for a products page like this with subitems that use different templates, you need to make sure that the rendering for each of those templates is set up correctly.

To set up the RSS feed rendering for a template:

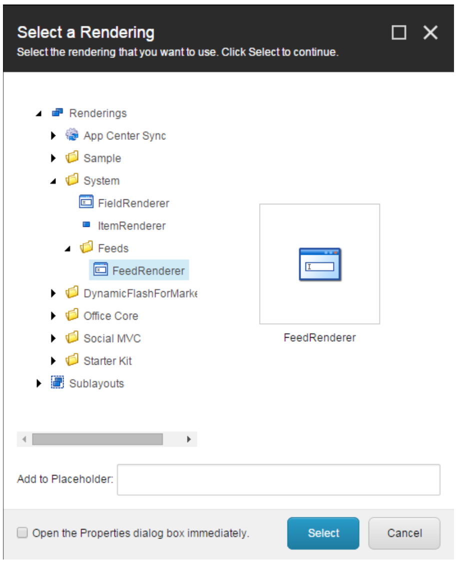
1. In the Content Editor, in the content tree, navigate to the relevant template. Alternatively, you can navigate to the subitem, identify the template that it is based on, and open the template from the item.
2. Expand the relevant template, and click the template's _Standard Values item.
3. On Presentation tab, in the Layout section, click Details.



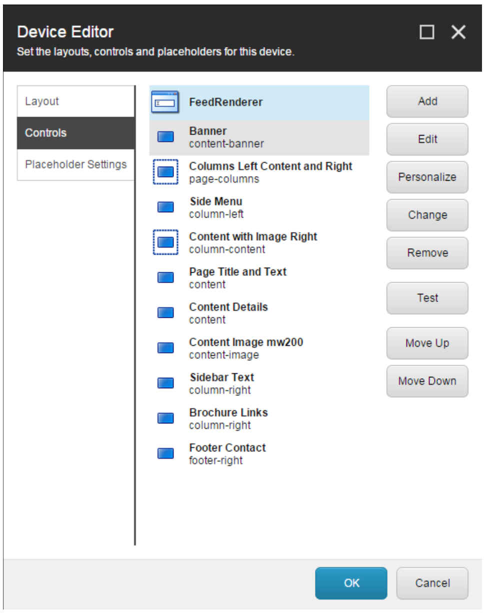
4. In the Layout Details dialog box, scroll down to the Feed section, and click Edit .
5. In the Device Editor dialog box, click Controls.



6. Click Add and in the Select a Rendering dialog box, click Renderings, System, Feeds, and then FeedRenderer.



7. Click Select and *FeedRenderer* now appears in the list of controls in the Device Editor dialog box.
8. In the Device Editor dialog box, to move *FeedRenderer* up to be the first item in the list, select the *FeedRenderer* item in the list and then click Move Up until it is at the top of the list.



9. Click OK and save your changes.

Now repeat this procedure for all the templates that are used for items under the parent item, in this case the *Products* item.

Send feedback about the documentation to docsite@sitecore.net.

Walkthrough: Setting up RSS feeds

In the Content Editor, you can set up RSS feeds for any item but, typically, you would set it up for items that are updated regularly, for example, news items and blogs.

This walkthrough describes how to set up an RSS feed for an item and outlines how to:

- Create a folder for the RSS feeds
- Create an RSS feed
- Configure the RSS feed
- Create a link on your website to the RSS feed

Create a folder for the RSS feeds

First you need to create a folder where you can store your RSS feed items.

To create a folder for the feeds:

1. In the Content Editor, in the content tree, select the Home item of the website.
2. On the Home tab, in the Insert group, click the drop-down arrow and click *Insert from template*.

3. Click the Browse tab and navigate to Templates/Common/Folder

4. Give the new folder a name — RSS Feeds.

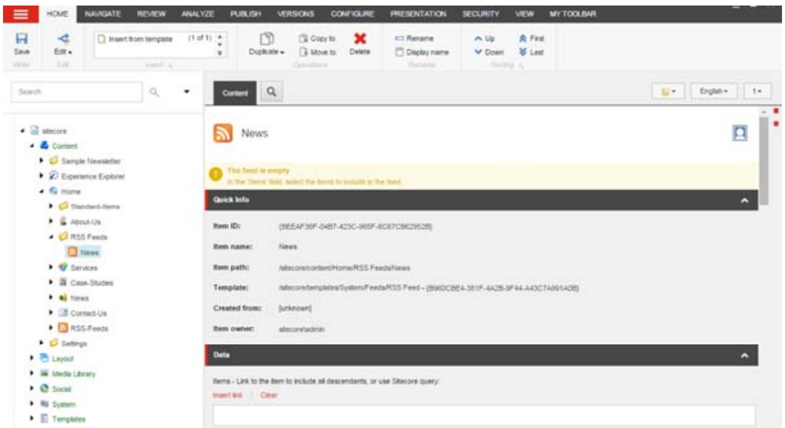
Create an RSS feed

Now that you have created a folder to store the RSS feeds in, you can create an RSS feed and link it to the item you want to include in the RSS feed.

To create an RSS feed in the Content Editor:

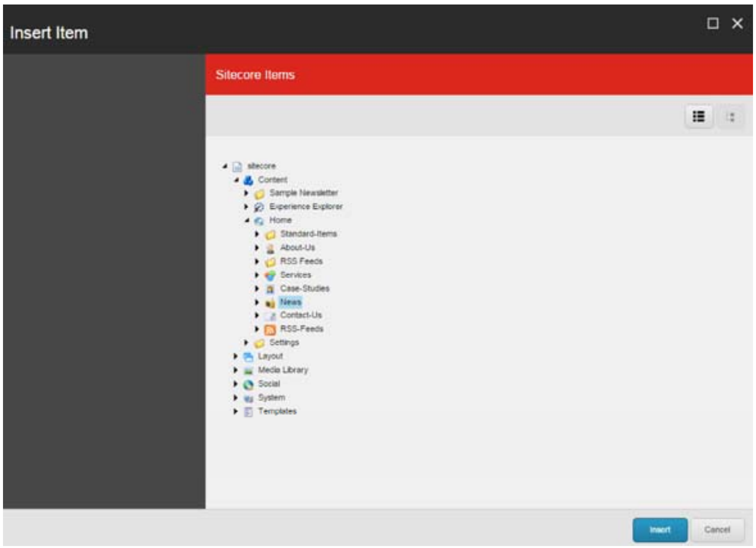
1. In the Content Editor, select the *RSS Feeds* folder that you just created.
2. On the Home tab, in the Insert group, click Insert from Template.
3. In the Insert from Template dialog box, navigate to Templates/System/Feeds/, and select *RSS Feed* to create a feed based on this template.

4. In the Item Name field, enter a name for the new feed.
5. Click Insert and the new feed item is created and opened in the Content Editor.



The new feed displays a warning informing you that the feed is empty and that you must select an item to include in the feed.

6. On the Content tab, in the Data section, for the Items field, click Insert Link.
7. In the Insert Item dialog box, click the item that you want to include in the RSS feed and then click Insert item.



Note

You can only create an RSS feed to an item that has subitems. If you create an RSS feed to an item that does not have any subitems, Sitecore tells you that the feed is empty.

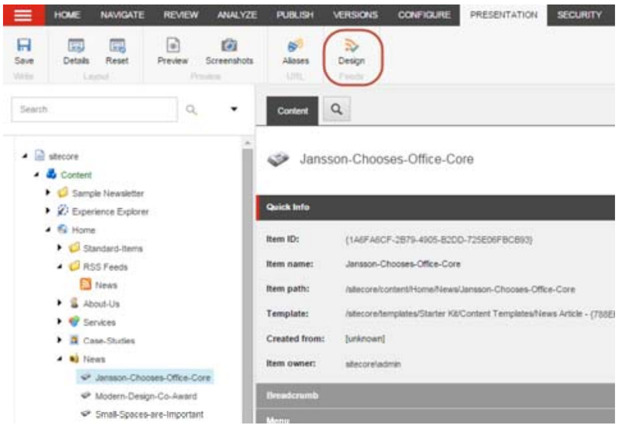
8. In the Description field, enter a description of the RSS feed.
9. Save the new feed item.

Configure the RSS feed

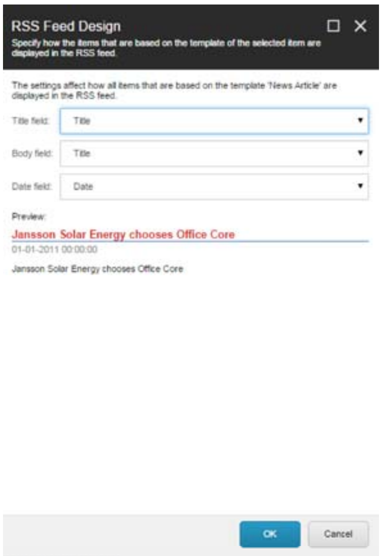
Now, you must configure how the feed information is presented in the RSS feed. The feed should not tell subscribers that the item is updated; instead, it should contain information about the information that are created or updated.

To configure how the RSS feed is presented:

1. In the Content Editor, expand the item that you linked the RSS feed to and click one of its subitems.
2. On the Presentation tab, in the Feeds group, click Design.



3. In the RSS Feed Design dialog box, specify how the feed is presented to the user who subscribes to this RSS feed.



Field	Value
Title	The item field that you want to appear in the title of the feed.
Body	The item field that you want to appear in the body of the feed.
Date	The date field that you want to appear in the feed. This could be, for example, the date the item was created, updated, or the date from which the news applies.
Preview	A preview of the feed based on the selections above.

4. When you have filled in these fields, click OK.

   On the Presentation tab, in the Feeds group, the Design button now has a green check mark that indicates that an RSS feed has been designed for this item.

   Important

   When you specify a design for an item, all the other items that are based on the same template are presented in the same way in the feed. If some of the subitems are based on a different template, you must also specify how those items should be presented in the feed and make sure to setup RSS feed rendering correctly.

5. In the Content Editor, save your changes.

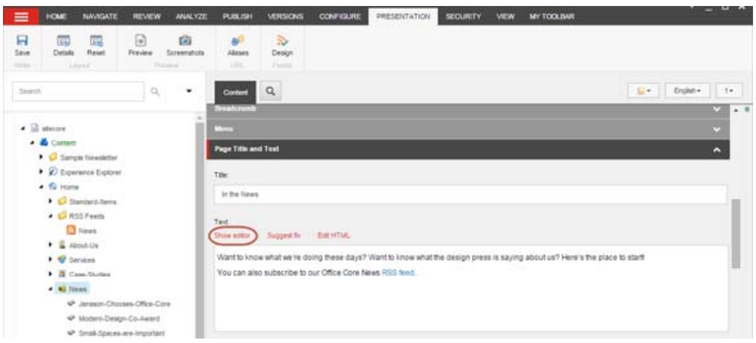
When you have configured how the feed should be presented, the RSS feed item should no longer display any warnings.

Create a link on your website to the RSS feed

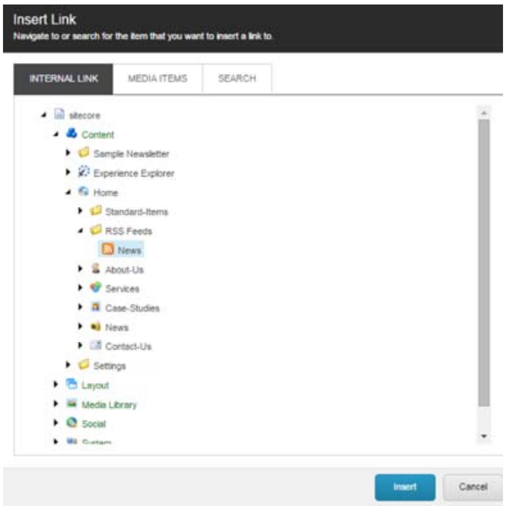
After you have configured the RSS feed, you must create a link to the feed in the parent item so that users can subscribe to the feed from your website.

To create a link to the content item:

1. In the Content Editor, open the item that you created an RSS feed for - for example, *News*.
2. In the content area, on the Content tab, expand the Page Title and Text section and click Show editor.



3. In the text editor, enter the text for the link, select the text and then click Insert Sitecore Link.
4. In the Insert a Link dialog box, on the Internal Link tab, navigate to the RSS feed item that you created earlier, or search for the RSS feed on the Search tab.



5. Click Link to create the link.
6. Save your changes.

Send feedback about the documentation to docsite@sitecore.net.

Move an item to the next workflow state

An item that is subject to a workflow, must go through all the workflows states before you can publish it. When you create a new item, it is automatically placed in the initial workflow state. The item remains in the initial workflow state until you use a workflow command to move it to the next workflow state.

Note

An administrator can overwrite any of these workflow states and move an item through the states or publish the item at any given time.

If you edit an existing item that is in a workflow, Sitecore creates a new version of the item and places it in the initial workflow state. If you edit this item again later, you edit the version that is in the workflow unless you specify that you want to edit another version.

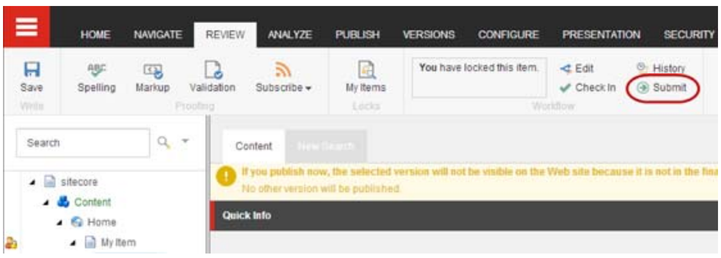
Edit an item and move it to the next workflow state

To send a content item to the next workflow state:

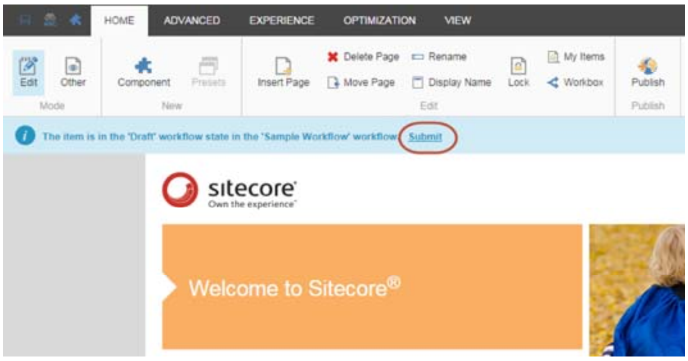
1. In the Content Editor or the Experience Editor, navigate to the relevant item or page.
2. Edit the item and save your changes.
3. If the item is in a workflow, you can see the available workflow commands:
   - In the Content Editor, on the Review tab, in the Workflow group.
   - In the Experience Editor, in the notification bar.

   In the following example, the workflow command *Submit* is displayed, and therefore this is the only workflow command that you can apply to this item in its current workflow state.

   The Content Editor



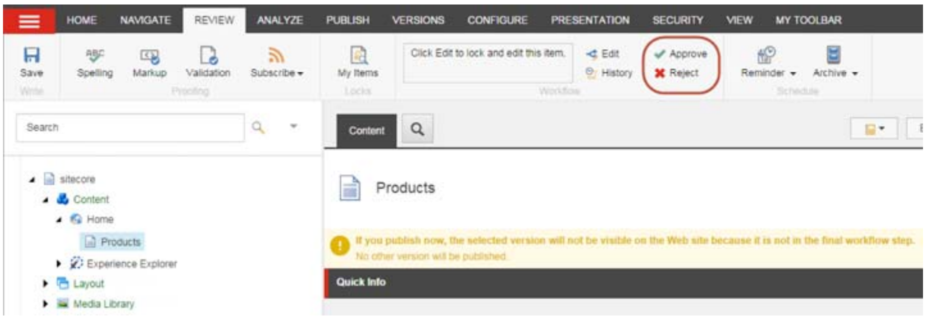
   The Experience Editor



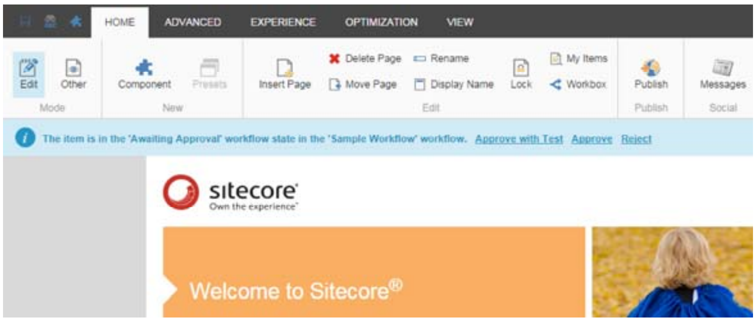
4. Click the relevant workflow command to send the item to the next workflow state.
5. In the dialog box that appears, you can enter a comment about the page you have edited, and click OK.

   The item is now in the next workflow state and new workflow commands are available in the notification bar. If you do not have permission to edit the item in the current workflow state, you cannot lock the item for editing and the workflow commands are not displayed.

   The Content Editor



   The Experience Editor



To get an overview of all the items that are currently in workflows, you can open the [Sitecore Workbox](#).

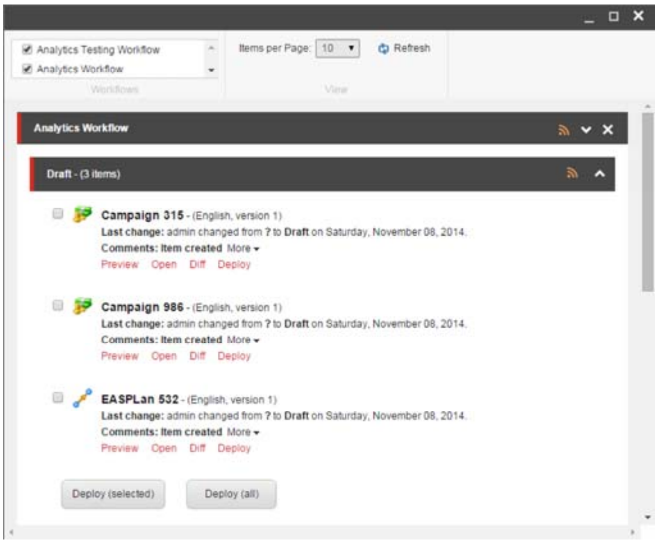
Send feedback about the documentation to docsite@sitecore.net.

Review an item in the Workbox

The Workbox is a tool for reviewing and managing items that are in a workflow. When you need to review your items, use the Workbox to see all the items that are currently in workflows and the workflow states that they are in.

To review an item in the Workbox:

1. Log in to the Sitecore, and on the Launchpad, click Workbox.
2. In the Workbox, in the Workflows group, select one or more of the workflows that you want to view to see all the relevant workflow states and the items that are in each state.



3. For each item in a workflow state, click one of the following options:
   - Preview, to preview the version of a content item that is in a workflow state. This opens the specific version of the item in a new browser window.
   - Open, to open the item in the Content Editor.
   - Diff, to compare the differences between two versions of the same item. In the Compare Versions dialog box, select the two versions that you want to compare and whether to compare them in one field or side by side in two columns.
4. When you have finished reviewing the item, you can apply one of the workflow commands that have been defined for the workflow state. For example, in the Awaiting Approval workflow state, you can approve or reject the item and thereby move it to the next appropriate workflow state.

Send feedback about the documentation to docsite@sitecore.net.

The Workbox commands

Before you publish a content item on your website, commands in the Workbox let you preview, open, or compare different versions of an item, as well as approve or reject the content.

Some commands can be executed on many items at the same time, and only certain commands are shown for all items, while others appear depending on the workflow commands configured for the corresponding workflow state.

In the Workbox, in the Workflows section, you can configure the Workbox to only show specific workflows.

The following table describes the commands for each item in the Workbox:

Command	Description
Preview	Preview the item in a new window.
Open	Open the item in a new window in the Content Editor.
Approve	Approve the item and move it to the next workflow state. When you approve the item, you are prompted to enter a comment. This text will be sent as a notification to the next reviewer.
Diff	Open the Compare Versions dialog box to display the difference between two versions of the same item.
Reject	Move the item to an earlier state in the workflow.
Deploy	Publish the item.
Approve with Test	Approve the item, move it to the next state in the workflow and launch an A/B test.
Approve without Test	Approve the item without performing an A/B test and move it to the next state in the workflow.
More	Show the editing history and details of the last workflow change made to the item.
Commands in the Batch Operations toolbar	Description

Approve (all)	Approve a batch of preselected items and move them to the next workflow state.
Reject (all)	Reject a batch of preselected items and move them back to an earlier stage in the workflow.
Deploy (all)	Publish a batch of preselected items.
Approve with Test (all)	Approve a batch of preselected items and run an A/B test.
Approve without Test (all)	Approve a batch of preselected items without running an A/B test.

Send feedback about the documentation to docsite@sitecore.net.

Workflows

To ensure that only items that have been approved are published, you can use workflows to formalize and enforce your company processes and procedures.

A workflow is a series of states that reflects the work processes and procedures for creating web content in your organization. For example, your workflow states can correspond to the creation, review, and approval stages that items must go through before they are published on your website.

A workflow can contain any number of workflow states and be as simple or complex as required. Items do not necessarily have to move through all the workflow states that you have defined — some states can be optional or only applicable to certain kinds of content.

Note

An administrator can overwrite any of these workflow states and move an item through the states or publish the item at any given time.

Furthermore, you can implement any number of different workflows. As a minimum, each workflow must contain two workflow states – one that is the initial workflow state and one that is the final workflow state. You can only publish an item in a workflow, if it is in the final workflow state.

Typically, developers implement workflows, but it can be useful for content authors to understand how a workflow is constructed and implemented.

The following workflow items are stored in the content tree:

- Workflows – the main item that contains all the subitems that make up the workflow.
- States – the basic building blocks of the workflow. The workflow states represent the different stages in your content creation process.
- Commands – the actions that must be performed by contributors to move the items through the workflow states.
- Actions – the actions that are automatically performed on the items when they are in particular workflow states or when particular workflow commands are executed.

An example of a workflow:

cms_WorkflowDiagram

After the workflow has been defined, the developer specifies the item templates that are subject to the workflow. Every time a content author creates or edits an item that is based on one of these templates, the item is sent to the initial workflow state of the workflow.

Send feedback about the documentation to docsite@sitecore.net.

Workflows and the Workbox

A workflow is a series of predefined states that reflect the work processes and procedures for creating web content in your organization. For example, your workflow states could correspond to the creation, review, and approval stages that items must go through before they are published on your website.

Workflows enable you to have control over the content approval and publishing process, ensuring that only content that has been approved is published.

Workflows consist of three elements – states, commands and actions.

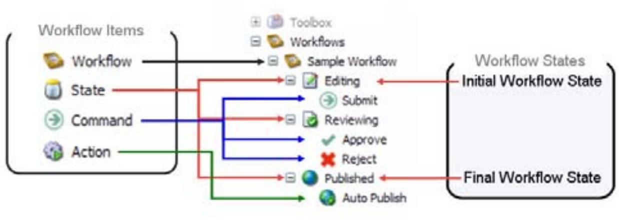
A workflow can contain any number of workflow states and be as simple or complex as required. Items do not necessarily have to move through all the workflow states that you have defined. Some states can be optional or only applicable to certain kinds of content.

The Workbox is a tool in Sitecore that displays information about the items in a workflow such as editing history or the number of items that you can find in each workflow state. In the Workbox you can find all items in a workflow.

Workflows

In Sitecore, you can implement any number of different workflows. As a minimum, each workflow must contain two workflow states – the initial and final workflow state. You can only publish an item in a workflow if it is in the final workflow state.

An example of a workflow:



In this example, the workflow consists of three states; *Editing*, *Reviewing* and *Published*.

All of the following workflow items are stored in the content tree:

- Workflows – the parent item that contains all the subitems that make up the workflow.
- States – the basic building blocks of the workflow. The workflow states represent the different stages in your content creation process.
- Commands – move content items from one workflow state to another. When the command is executed, the item is moved to the workflow state that is defined in the command item.
- Actions – the actions that are automatically performed on the items when they are in particular workflow states or when particular workflow commands are executed.

- Note

  An administrator can overwrite any workflow state and move an item through the states or publish the item at any given time.

The Workbox

The Workbox gives you an overview of all items that are in workflows and enables you to manage these items.

In the Workbox, you can see the editing history of items in the workflow, each item's workflow state, and the number of items in each workflow state.

For each item in a workflow state, you can preview, open and [compare different versions] of the same item. You can also set up an RSS feed, which enables you to receive a notification when there are changes to an item or when a version of an item enters or leaves a specific workflow state.

You can open the Workbox from the Launchpad.

Send feedback about the documentation to docsite@sitecore.net.

Media items

A media item is composed of several fields and it always has a file attached to the Media field. A media item can be an image, a document, a video, or an audio file.

In Sitecore, there is a difference between media files and media items. An image or video file that is stored on your computer is called a media file. When you upload the media file to the Sitecore Media Library, the file is attached to a media item and is from then on referred to as a media item.

A Media item can be:

- Images — .jpg, .gif, .png and other formats.
- Microsoft Word documents — .doc and .docx files.
- PDF documents
- Videos — .MPEG, .WMV and other formats.
- Audio files — .MP3 and other audio formats.

You should store and manage your media items in the Media Library. To edit your media items, you can use the Sitecore tool the Image Editor for images or other applications that you have installed on your computer such as Photoshop or Microsoft Word for documents. You can open the Image Editor from the Sitecore Start menu.

Send feedback about the documentation to docsite@sitecore.net.

The Media Library

You can manage all your media items from the Media Library, such as images that you want to embed in a webpage or make available for download by visitors. The Media Library contains all your media items, such as images, documents, videos, and audio files.

In the Media Library you can:

- Keep all your media files in one place and organize them in a folder structure similar to the content tree.
- Navigate the content tree or use Sitecore's search functionality to quickly find media items.
- Upload media files to the Media Library. You can use the upload buttons to browse your computer for images or simply use the Drag & Drop dialog box to add media files.
- Make every media content item and media file versionable.

Note

To edit your images, you can use Sitecore's Image Editor.

This topic outlines how to:

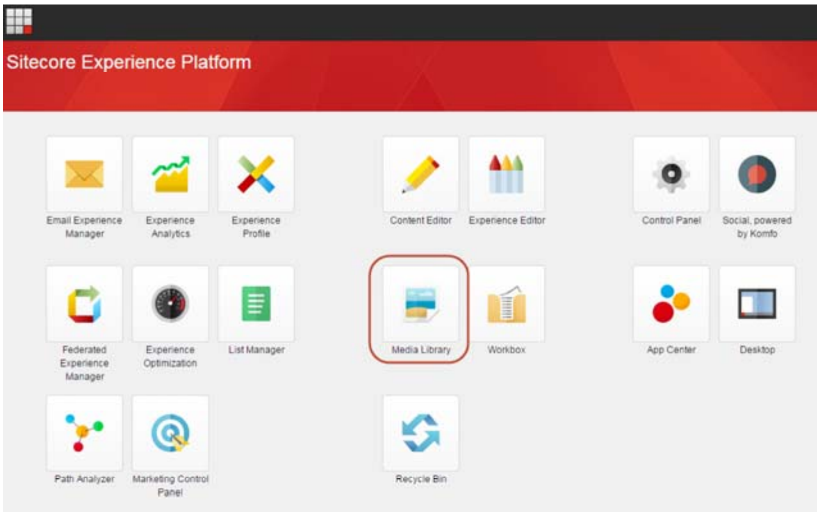
- [Open the Media Library]
- [Navigate the Media Library user interface]

Open the Media Library

You can open the Media Library from the following places.

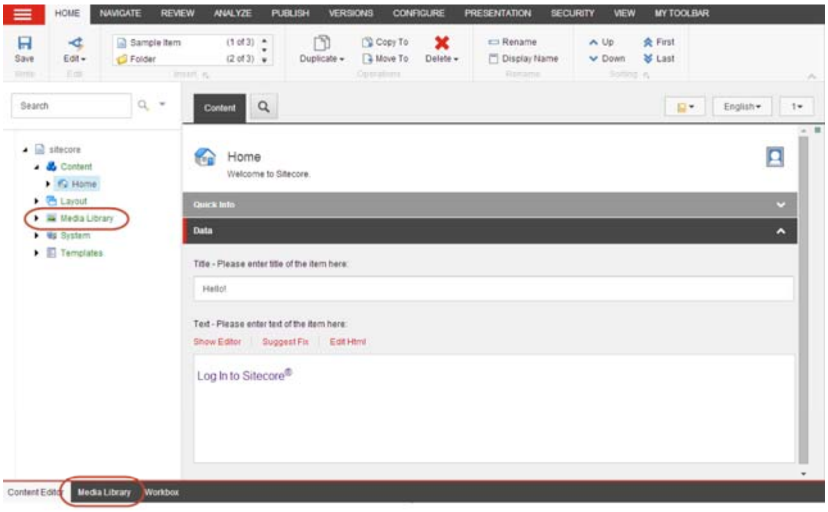
The Launchpad

To navigate to the Media Library, on the Sitecore Launchpad, click the Media Library icon.



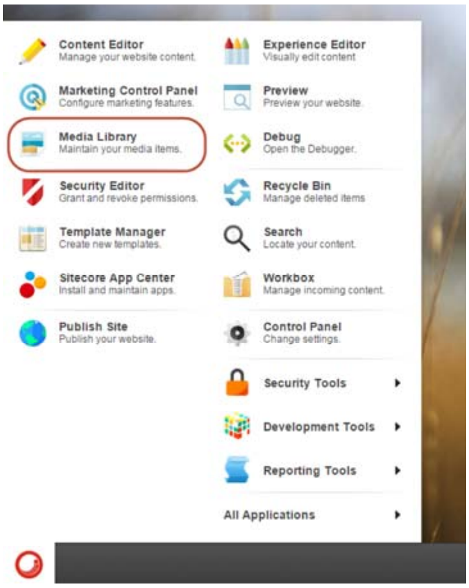
The Content Editor

In the Content Editor, navigate to the Media Library in the content tree. If you want to open the Media Library separately, click Media Library in the page bar at the bottom of the Content Editor.



The Sitecore Desktop

To open the Media Library, click the Sitecore Start button and then click Media Library.



Navigate the Media Library user interface

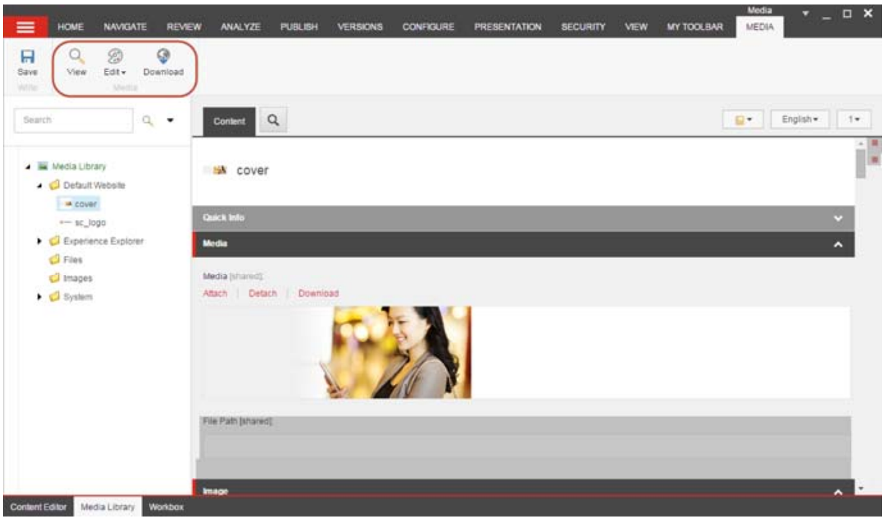
The Media Library user interface consists of:

- The ribbon
- The content tree
- The content area

The ribbon

In the Media Library, you get the same set of options on the ribbon as you do in the Content Editor, although the Insert group allows you to create new media folders and upload media files.

If you select a media item in the content tree, the Media tab displays in the ribbon. This contains the new Media group with the actions that you can perform on the media item.



Note

The functionality in the ribbon depends on the user rights you have been assigned by the website administrator and the type of media item you have selected.

The content tree

Use the content tree or the search to find media items. Expand folders in the content tree to see all subfolders and media items and collapse the folders again to hide subfolders.

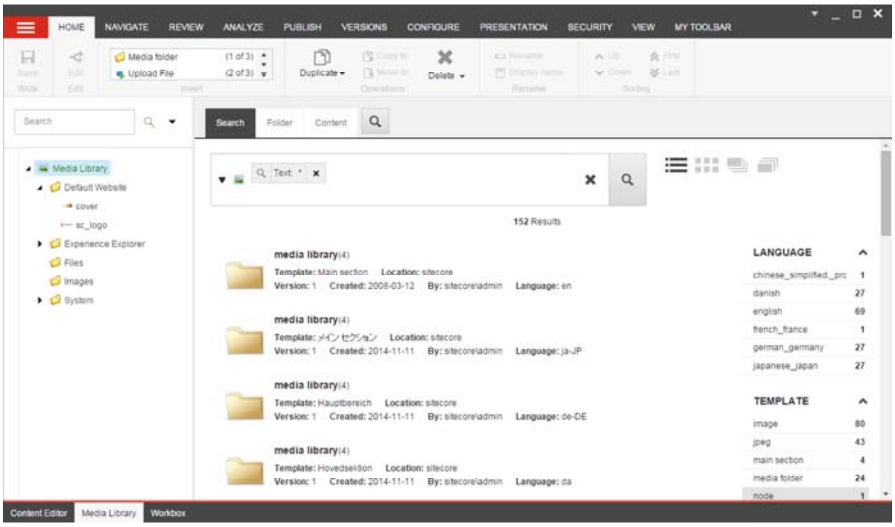
Make sure you store all your media items in folders and give the folders meaningful names. An easy way to organize your Media Library folder structure is to mirror the structure of your website. For example, if your site has four main sections: *Home*, *Products*, *Jobs*, and *Contact Us* you could create four folders with the same names to store all your images and other media. This will make it easier to locate media items as your site grows and you need to update media items.

The content area

The content area displays the following tabs that you can use when working with a specific media item.

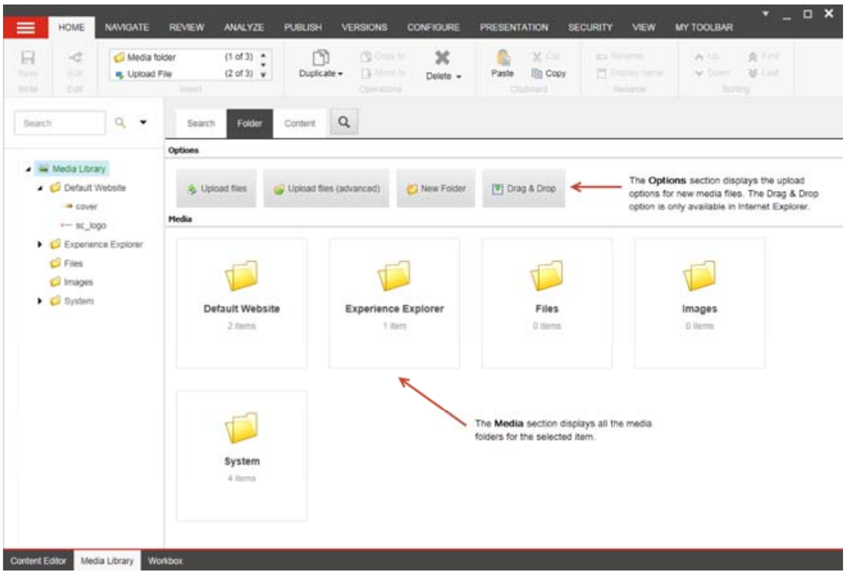
The Search tab

This tab offers an elaborate search functionality that can help you find very specific items among maybe thousands of items in the Media Library. The Media Library opens by default on the Search tab, with a search already run for all items in the Media Library.



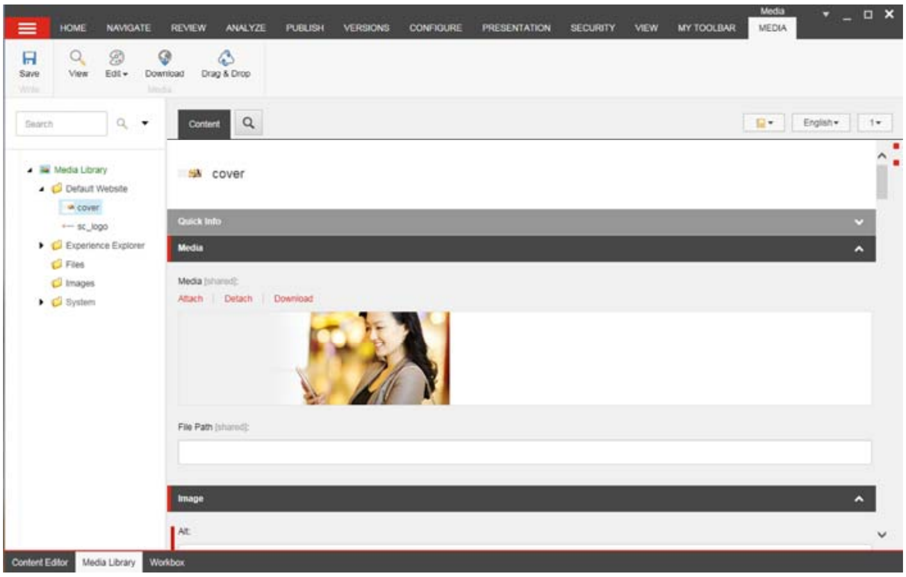
The Folder tab

This tab displays the options and media contents for the selected folder in the Media Library.



The Content tab

This tab displays all the fields that you can edit for the media item that you have selected.



Send feedback about the documentation to docsite@sitecore.net.

Edit a media item

In the Media Library, you can edit media items such as images, documents and videos either using Sitecore tools such as the Image Editor for images or with other applications that you have installed on your computer such as Microsoft Word for documents or Photoshop for images.

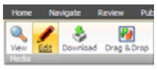
This topic outlines how to:

- Edit a media item from the Media Library
- Open and edit an image from the Image Editor
- Edit an image in the image field

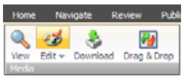
Edit a media item from the Media Library

To edit a media item from the Media Library:

1. In the Media Library, navigate to the item that you want to edit.
2. On the Media tab, in the Media group, click Edit. The icon changes depending on the type of media item selected. For example:
   - If the media item is a Word document, you see a pencil as the associated icon. This opens the document in Microsoft Word.



   - If the media item is an image, you see an artist's palette. This opens the image in the Sitecore Image Editor.



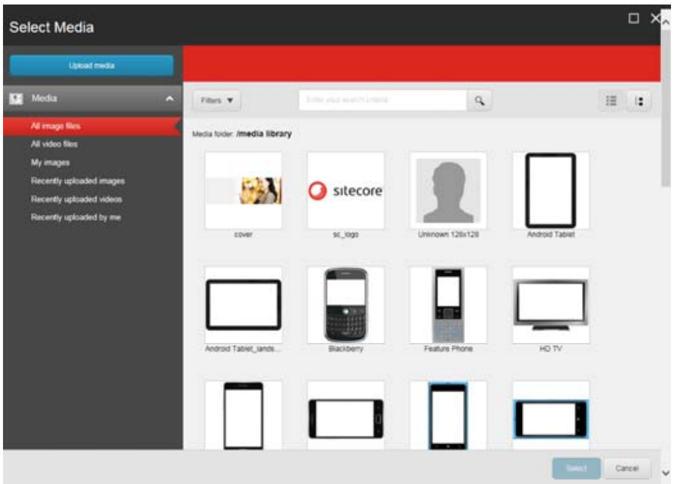
3. When you have finished editing your media item, save your changes and close the editor.
4. In the Media Library, click Save.

Open and edit an image from the Image Editor

To open an image for editing in the Image Editor:

1. Click the Sitecore Start menu.
2. Click All Applications and then click Image Editor.
3. In the Image Editor, in the ribbon, click Open.
4. In the Select Media dialog box, select the image that you want to edit and click Select.

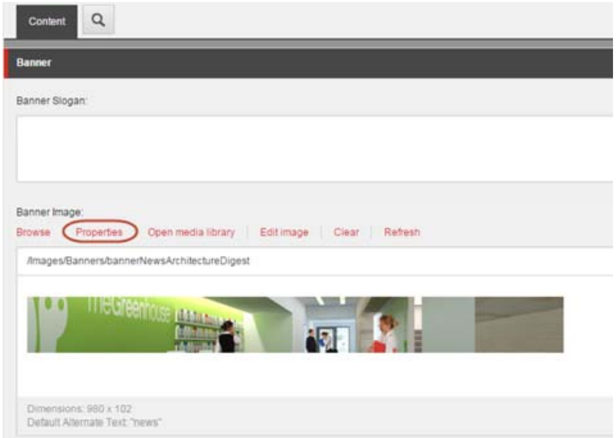
   You can find the image that you want to edit in the left menu, or you can use the filters or the search functionality. Alternatively, you can click Upload media to upload a new image.



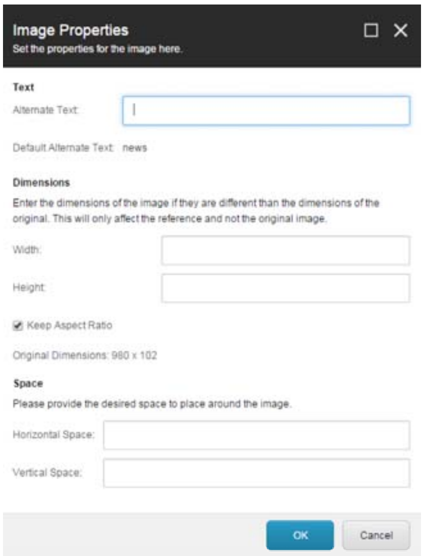
Edit an image in an image field

To edit an image in an image field:

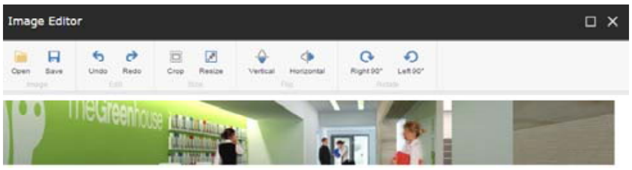
1. In the Content Editor or the Experience Editor, navigate to the item or page that you want to edit. To edit the image properties:
   - In the Content Editor, on the Content tab, scroll to the field that contains the image and click Properties.



   - In the Experience Editor, click the image that you want to edit, and in the floating toolbar that appears, click Modify Image Appearance.
2. In the Image Properties dialog box make the relevant changes to:
   - The Alternate Text field - enter the text that will appear when the image cannot be displayed.
   - The Dimensions section - specify the dimensions that the image should be displayed in, if they differ from the actual size of the image.
   - The Space section - specify how much space should be placed around the image.



3. To edit the image itself, you must open the image in the Image Editor, where you can crop, resize, flip, and rotate the image. To open the Image Editor:
   - In the Content Editor, on the Content tab, scroll to the field that contains the image and click Edit image.



   - In the Experience Editor, click the image that you want to edit, and in the floating toolbar that appears, click More and then Edit the related item. This opens the item in the Content Editor, from where you can scroll to the field that contains the image and click Edit image.

   Note

   You can also edit the image with any image editor that is available on your local computer.

4. When you have finished with editing the image, save your changes.
5. To see your changes to the image:
   - In the Content Editor, save your changes.
   - In the Experience Editor, click the System menu, and then Exit to return to the Experience Editor.

   Important

   All the changes that you make are applied to the version of the image that is stored in the Media Library and not just the copy that is shown on your web page.

Send feedback about the documentation to docsite@sitecore.net.

Insert an image

In the Content Editor or the Experience Editor, you can insert various types of media, typically images, into an image field, a Rich Text field, or a Word field. These media items can be images, animations, film clips, sound files, and so on.

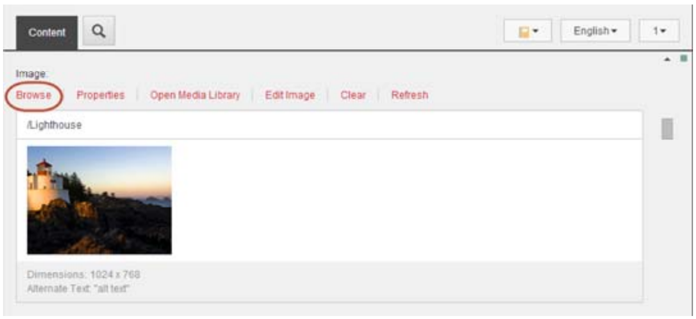
This topic outlines how to:

- Insert an image into an image field
- Insert an image into a text field

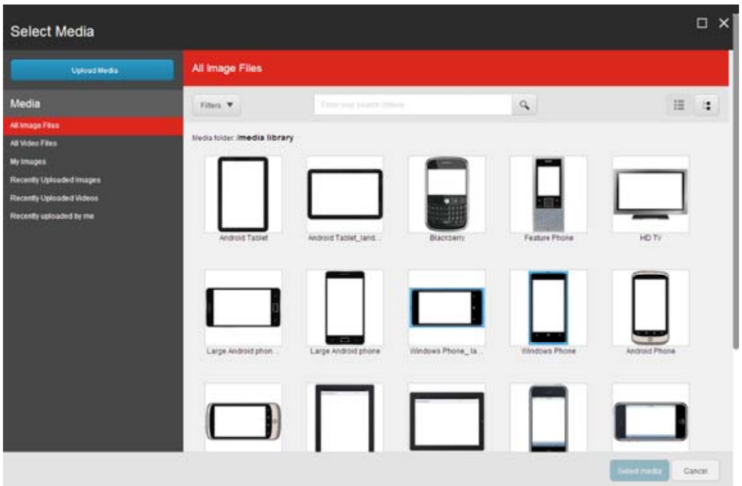
Insert an image into an image field

To insert or change an image in an image field:

1. In the Content Editor or Experience Editor, navigate to the item or page that you want to edit:
   - In the Content Editor, click the Content tab, locate the image field where you want to insert an image and click Browse.

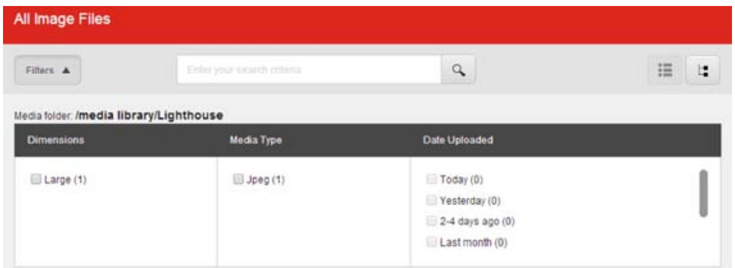


   - In the Experience Editor, click the image field where you want to insert an image and in the floating toolbar that appears, click Choose an Image.
2. The Select Media dialog box displays all the items from the Media Library that is available for the image field you have chosen.



   The items in the Select Media dialog box are not listed in any particular order, so use the search functionality or the predefined lists from the menu to find your image:

   - In the Media menu, use the links to quickly get a complete list of All image files, All video files, My images, Recently uploaded images, and Recently uploaded videos.
   - To filter the displayed media items according to specific metadata of the image, click the Filter drop-down arrow and select the required filters. The number in parenthesis indicates how many items are available for the filter.



   - To search for a specific item, enter a search word in the Search field.
   - To navigate through the content tree to find the image that you want to use, click Tree View in the upper right corner. To go back to the list view, click List view.
3. Select the image that you want to use, click Select Media, and the new image is inserted into the image field.

   Note

   If the image that you want to use is not available, click Upload media to upload a new image to the Media Library.

Insert an image into a text field

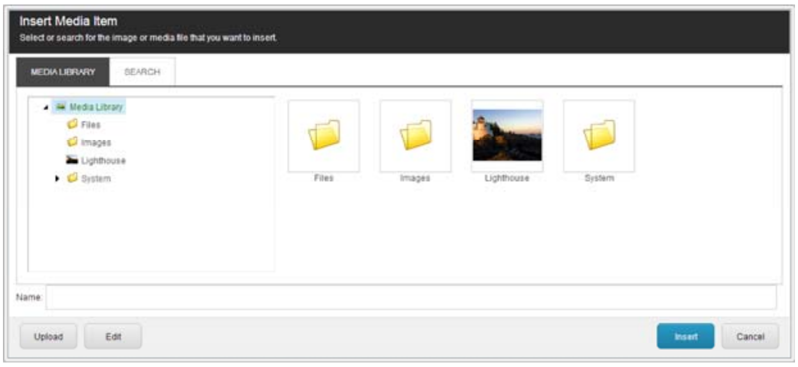
To insert a media item into a text field:

1. In the Content Editor or the Experience Editor, navigate to the text field, where you want to insert a media item.
2. You can insert the image in one of the following fields:
    - To insert the image in the Rich Text Editor, open the editor from a Rich Text field and click Insert Sitecore Media.
    - To insert the image in a Word field, open Microsoft Word from a Word field and click Insert Media.
    - To insert the image in a Rich Text field directly on the page in the Experience Editor, click the field and in the floating toolbar, click Insert Image.

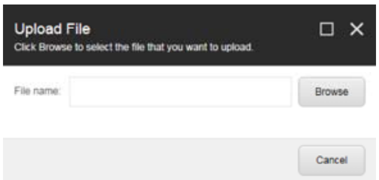
    Note

    If you do not see the Insert Image button in the floating toolbar, click the More drop-down arrow.

3. If the image is already in the Media Library, in the Insert Media Item dialog box, you find and select the image that you want to insert and then click Insert. To locate the image you can navigate through the content tree on the Media Library tab, or you can search for the item on the Search tab.



4. If the image is not available in the Media Library, click Upload.
5. In the Upload a File dialog box, click Browse and then select the image file that you want to use.



6. In the Open dialog, click Open to upload the image file to the Media Library.
7. The image file opens in the Content Editor. To save the new image and return to the Insert Media Item dialog box, click Save/Close.
8. To insert the image in the text field, click Insert.

    Note

    If you click Cancel, the Insert Media Item dialog box closes, but the new item remains in the Media Library.

Send feedback about the documentation to docsite@sitecore.net.

Upload a file to the Media Library

In the Media Library, there are three different options to upload your media files:

- Upload files – use this option to upload a single file or multiple files from the same location.
- Upload files (advanced) – use this option to apply certain upload options. For example, you can upload a Zip archive or overwrite duplicates.
- Drag & Drop – use this option to upload a single file or multiple files by dragging them from Windows Explorer to the Sitecore Media Library.

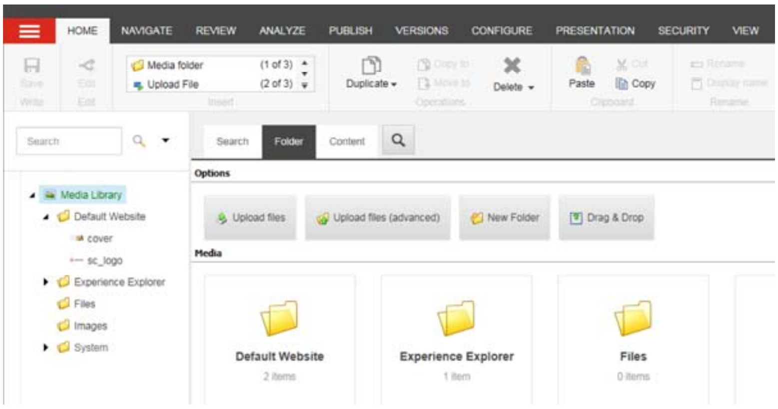
    Note

    The drag-and-drop functionality is currently only supported in Internet Explorer 7 or later. If you use a different browser such as Mozilla Firefox, you must use the Upload files and Upload files (advanced) options to upload your files to the Media Library.

Upload files

To upload a single media file:

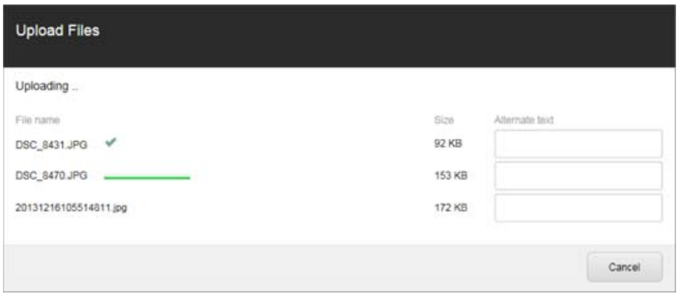
1. In the Media Library, in the content tree, select the folder where you want to upload your media files to.
2. In the editing pane, on the Folder tab, click Upload files.



3. In the Select file(s) to upload dialog box, browse to and select the media file that you want to upload and click Open.

   Note

   To select multiple files, press and hold the CTRL key when you select the files.



In the editing pane, on the Folder tab and in the Media section, you can see the images that you uploaded. The new items may contain a warning that you can remove, if you open the media item and enter an alternate text in the Alt field.

Upload files (advanced)

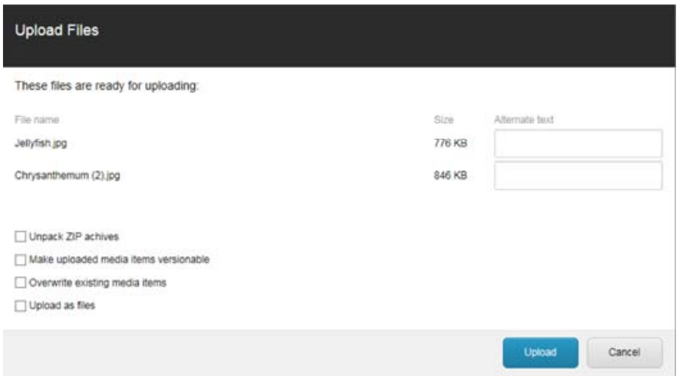
To upload files and apply certain upload options:

1. In the Media Library, in the content tree, select the folder that you want to upload your media files to.
2. In the editing pane, on the Folder tab, click Upload files (advanced).
3. In the Select file(s) to upload dialog box, browse to the folder on your computer that contains the images you want to upload.

   Note

   To select multiple files, press and hold the CTRL key when you select the files.

4. Click Open.
5. In the Upload Files dialog box, enter an alternate text for each media file and select the relevant upload option check boxes.



Upload option	Description
Unpack ZIP archives	Select this option to upload and extract a Zip archive to the Media Library. In the .zip file, you can include subfolders with files. The folder structure is re-created in the Media Library after the files are extracted.
Make uploaded media items versionable	Select this option to make the media file versionable. This creates the media file in the selected language only, and any other language version that you want you need to create manually.

Overwrite existing media items	Select this option to upload a file with the same name as a file that already exists in the Media Library, and want to overwrite the existing file. If you do not select this option, a duplicate file is created in the Media Library.
Upload as files	Select this option to store the file in the file system rather than in the database.

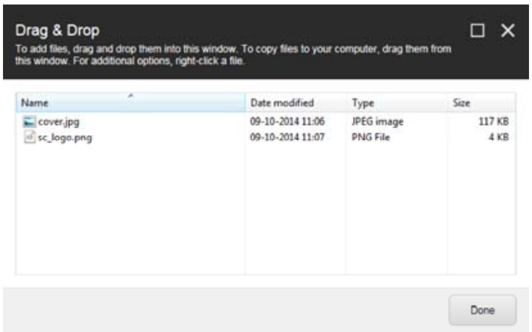
6. Click Upload to upload the images to the selected folder in the Media Library.

Drag & Drop

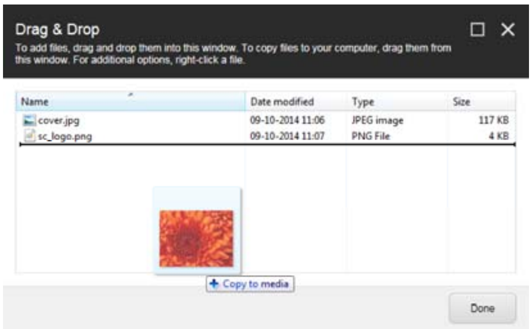
If you work in Internet Explorer 7 or later, the drag-and-drop option is the quickest and easiest way to upload media files to the Media Library. You can upload single or multiple files from your computer to the Drag & Drop dialog box. This is especially useful when you want to upload many files to different locations in Sitecore or from different locations on your computer.

To drag and drop files to the Media Library:

1. In a Media Library, select the folder that you want to upload the files to.
2. To open the Drag & Drop dialog box, from the editing pane, on the Folder tab, click Drag & Drop.



3. Open Windows Explorer and navigate to the folder where the images are stored and select the images you want to upload.
4. Position the Windows Explorer window next to the Drag & Drop dialog box and drag the selected images from Windows Explorer to the Drag & Drop dialog box.



Note

If you need to upload files to or from other locations, you simply navigate through the folder structure either in the Drag & Drop dialog box or in Windows Explorer. You do not have to close the dialogs and start over.

5. When you have dragged and dropped all your media files, click Done to copy them to the Media Library.

Note

If the folder you copy to already contains a media items with the same name, the Copy File dialog box opens, asking you to decide how to handle the duplicate media items.

In the editing pane, on the Folder tab and in the Media section, you can now see the images that you uploaded and that each new item contains a warning.

6. To remove these warnings, open each media item and enter the alternate text in the Alt field.

Make uploaded media files versionable by default

To make all the media files that you upload to the Drop & Drag dialog box versionable by default:

1. Open the web.config file that is stored in the web root folder of your website: C:\Inetpub\wwwroot\YourSitecoreWebsite\Website
2. Right-click the web.config file, click Open With, and then click Notepad.
3. Locate the following setting and change the value to “True”:

   <setting name="Media.UploadAsVersionableByDefault" value="false" />

4. Save your changes and close the web.config file.
5. Open the Media Library.
6. Click the Sitecore system menu and then click Application Options.
7. In the Application Options dialog box, click the Media tab, and select the Show only latest version of files option.

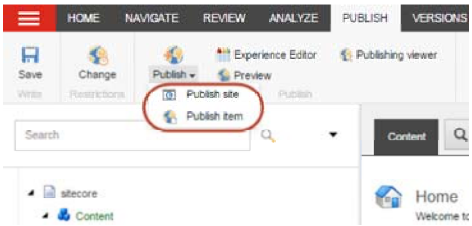
Send feedback about the documentation to docsite@sitecore.net.

Publish a website

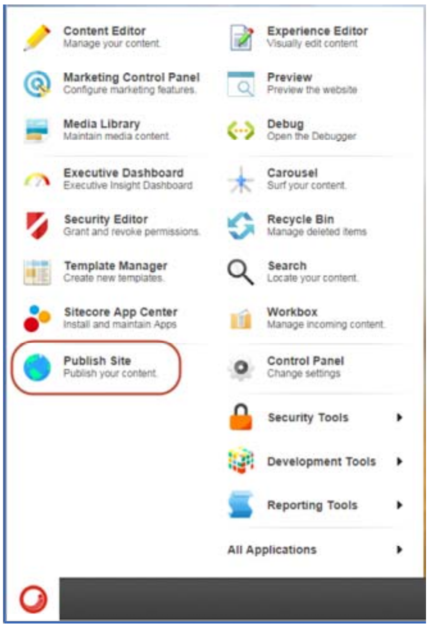
When you are ready to publish your website, you can run the publishing wizard, which guides you through the steps of publishing in Sitecore. You can run the publishing wizard from the Content Editor and the Sitecore Desktop.

To publish the entire website:

1. Open the Publish Site wizard from one of two locations in Sitecore:
   - From the Content Editor – on the Publish tab, in the Publish group, click the Publish drop-down arrow, and select Publish site.



   - From the Sitecore Desktop – click Sitecore Start, and then click Publish site.



2. In the Publish site wizard, specify the publishing mode for the website you are publishing:
   - Incremental Publish – publishes only items that are in the publishing queue. Every time you change an item, the item is automatically added to the publishing queue. If the changed item is part of a workflow, the item is added to the publishing queue when it reaches the final workflow state.

     Incremental Publish is the fastest way of publishing because Sitecore does not use resources to compare versions of the items that are in the publishing queue before publishing them.

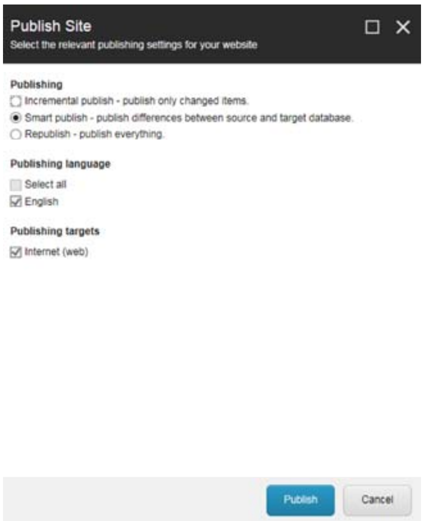
     Note

     Incremental Publish is only available when you publish the entire website.

   - Smart Publish – publishes all items that have changed since the last publication. When you run a Smart publish, every item in the Master database is compared with the equivalent item in the target database and, if an item has changed, it is published

     Comparing all items in the database makes this a time-consuming way of publishing your entire website, particularly if the content tree contains many items.

   - Republish – publishes everything. Republish overwrites every item in the target database with the equivalent item from the Master database, even if it has not changed. In this way, republishing removes all obsolete versions from the target database. You can use republishing to bring a new web database online, to restore a backup of the master database, and to add a new content language, a new publishing target, or other system items to the website.



3. In the Publishing language section, specify the language versions that you want to publish.
4. In the Publishing targets section, specify the publishing targets that you want to publish to.

   Note

   If you do not specify a publishing target, the website is published to all the publishing targets.

5. Click Publish. When the publishing completes, you can see how many items were processed during publishing and the number of items created, deleted, or updated on the target database.

Send feedback about the documentation to docsite@sitecore.net.

Publish an item to your website

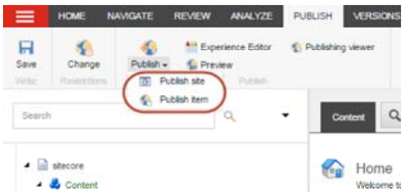
When you are ready to publish an item to your website, you can run the publishing wizard, which guides you through the steps of publishing in Sitecore. You can run the publishing wizard from the Content Editor and the Experience Editor.

Note

If you publish a new child item individually, the associated parent item must be published first.

To publish a single item:

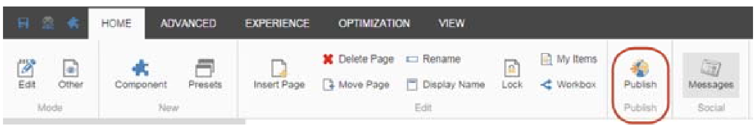
1. Open the Publish Item wizard from one of two locations in Sitecore:
   - From the Content Editor – select the item that you want to publish. On the Publish tab, in the Publish group, click the Publish drop-down arrow, and select Publish item.



   Note

   To publish the selected item in all languages to all publishing targets and avoid going through the steps of the publishing wizard, click Publish.

   - From the Experience Editor – navigate to the page that you want to publish, and on the Home tab, in the Publish group, click Publish.



2. In the Publish Item wizard, on the Settings page, specify the publishing mode:
   - Smart publish – publishes the selected item if it has changed since the last publication. When you run Smart Publish, the selected item in the Master database is compared with the equivalent item in the target database and, if the item has changed, it is published. When you are publishing single items, Smart Publish is a quick way to make sure that all changes to a specific item and its subitems are published.
   - Republish – publishes everything. Republish overwrites the selected item in the target database with the equivalent item from the Master database – even if it has not changed. In this way, all obsolete versions are removed from the target database. You can use republishing to restore a backup of the master database and to add a new content language, a new publishing target, or other system items.

3. To publish all the subitems of the selected item, select Publish subitems.
4. To publish all the related items to the selected item, select Publish related items. It is generally recommended that you select this option. Related items can be:
   - Clone references – items that are clones of the selected item.
   - Media references – media items that are related to the selected item.
   - Alias references – items that are aliases of the selected item.
5. In the Publishing language section, specify the language versions that you want publish.
6. In the Publishing targets section, specify where you want to publish the items to.

   Note

   If you do not specify a publishing target, the item is published to all the publishing targets.

7. Click Publish. When the publishing completes, you can see how many items were processed during publishing and the number of items that were created, deleted, or updated on the target database.

Send feedback about the documentation to docsite@sitecore.net.

Publishing

When you edit or create new items in Sitecore, they must be published before they appear on your website. This lets you save unfinished items and decide exactly when the items are launched to the website.

When you work in Sitecore, the items you save are saved to the Master database. When you publish new items or updated items, the items are copied from the Master database to a publishing target, and from the publishing target, the website is launched. The default publishing target in Sitecore is the Web database.

You can choose to publish the entire website or a single item:

- Site publishing – publishes your entire website starting from the root of the content tree. You can publish a site from the Content Editor or from the Sitecore Desktop.
- Item publishing – publishes the item you select in either the Content Editor or the Experience Editor. The item can only be published if all its ancestors have been published. With item publishing, you can choose to include all of the selected item's subitems and related items.

Send feedback about the documentation to docsite@sitecore.net.

Preview a webpage

The preview functionality lets you preview all your items on your website without publishing them first.

The preview functionality lets you:

- Specify a preview date, so that you can view your website in the past or future, according to the publishing restrictions that are set up.
- See how a webpage looks in different language versions.
- Preview how the page displays on different devices.

This topic outlines how to:

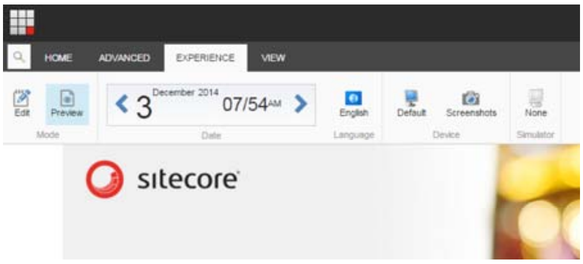
- Preview a page on your website
- Preview a page in the Device Simulator

Preview a page on your website

You can open the preview mode from either the Content Editor or the Experience Editor.

To preview a webpage:

1. To open the preview:
   - From the Experience Editor, on the ribbon, in the Mode group, click Preview.
   - From the Content Editor, on the Publish tab, in the Publish section, click Preview.
2. On the Experience tab, select the preview parameters:
   - Date – to specify another date to preview the webpage, in the Date group, click the date, or click the blue arrows to select the previous day or the next day.
   - Language – to view the page in a different language, in the Language group, click Language.

   Note

   The Experience tab might not be available to you if you do not have the necessary security permissions.
3. Navigate through the pages to preview your whole website.

   Tip

   In the preview mode, you can also use the search functionality to navigate to a specific page that you want to preview.

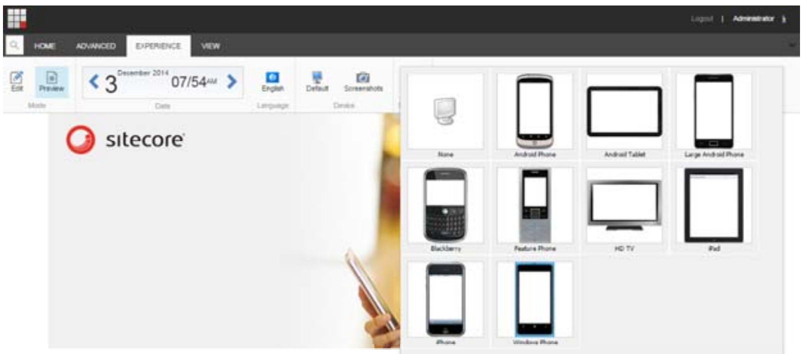
Preview a page in the Device Simulator

The Device Simulator enables you to preview your website in a simulation of several different devices, such as an iPhone, iPad, Android Phone, and Blackberry.

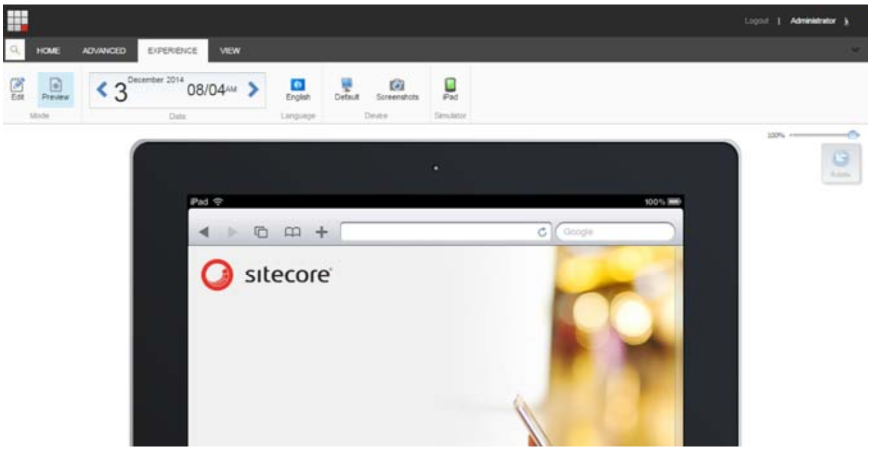
The Device Simulator mimics the devices and the preview is not a pixel-perfect rendition of the website as it appears on each device. However, it gives you a good indication of how the device presents the website and it lets content authors quickly see how pages look on mobile devices.

To preview a page in the Device Simulator:

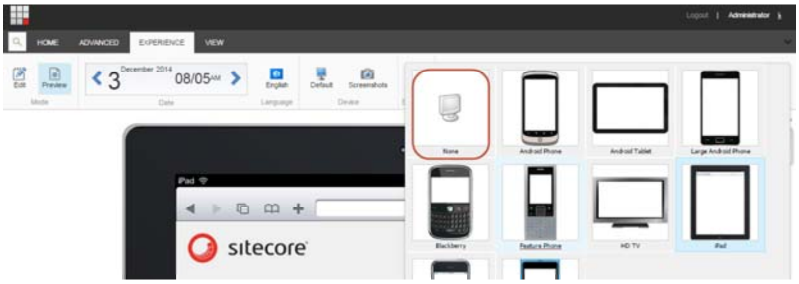
1. In the Experience Editor, on the Experience tab, in the Mode group, click Preview.
2. In the Simulator group, click the simulator button and select a device.



3. In the Device Simulator, all the navigation remains fully functional. You can:
   - Browse through the website.
   - Click Rotate to change the orientation of the device.
   - Scroll up and down, as well as left and right.



4. To return to the default page preview, click the simulator button again and in the drop-down menu, click None.



Send feedback about the documentation to docsite@sitecore.net.

Set up publishing restrictions for an item

In Sitecore, you can restrict the publication of items using publishing restrictions. Publishing restrictions let you specify that a specific item or a version of an item is publishable or unpublishable, either within a specific date range or indefinitely. This lets you control which items and pages are available for publishing at any given time.

For example, if you have set up Sitecore to periodically publish your entire site to make sure it is always up to date, you can create publishing restrictions on the items that you do not want to publish. This can be items that are still being created or finished items that you only want to publish when a certain campaign is rolled out. With publishing restrictions, you can easily manage when items are published, so that only the appropriate items are published.

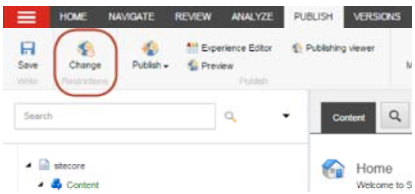
This topic outlines how to set up:

- Publishing restrictions for a single version of an item
- Publishing restrictions for all versions of an item

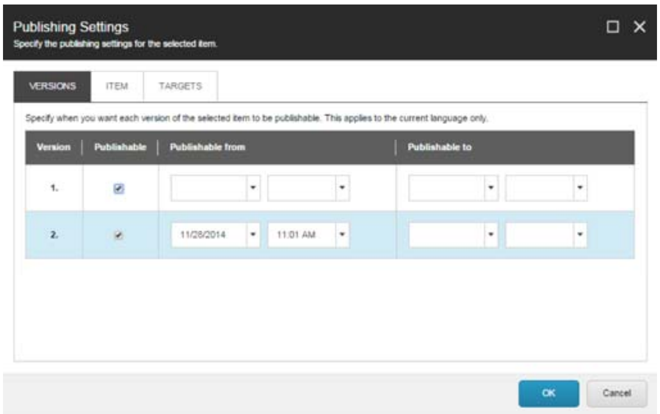
Publishing restrictions for a single version of an item

To set up publishing restrictions for a version of an item:

1. In the Content Editor, in the content tree, navigate to the item that you want to set up publishing restrictions for. Make sure that you select the relevant language and version of the item.
2. Click the Publish tab and in the Restrictions group, click Change.



In the Publishing Settings dialog box, on the Versions tab, the item versions in the current language are listed. By default, each version is publishable. If an item is not publishable, the item is not published, even if you publish the entire website.



Note

If an item is in a workflow, even if you have selected the Publishable check box in the Publishing Settings dialog box, the item is not publishable until it reaches the final workflow state.

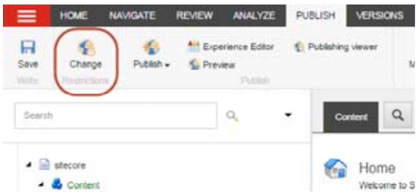
3. To make the specific item version publishable, select the Publishable check box. To make the item version not publishable, clear the Publishable check box.
4. To specify a period for when the item version is publishable, select the Publishable check box, and then enter the dates and times in the Publishable From and Publishable To fields. You can choose to specify only the start date, the end date, or leave all the fields blank, in which case the item version is publishable from the current day until you change it.
5. When you have finished setting up the publishing restrictions, click OK. In the Content Editor, a warning appears in the content area for the item informing you or other content editors that restrictions are set up for this item.

After you have set up the publishing restrictions for your items, you can preview your website in the Experience Editor to make sure that you have set up everything correctly. In the Experience Editor, you can browse through the pages on your website and see how they look on upcoming dates or in different languages.

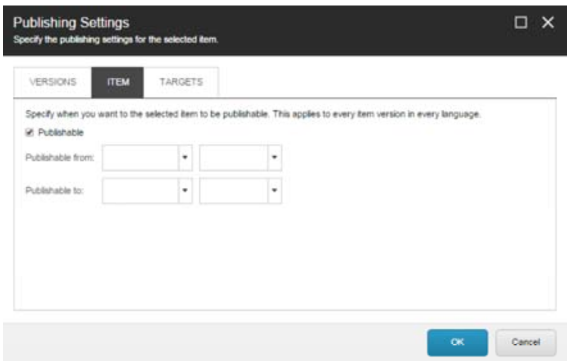
Publishing restrictions for all versions of an item

To set up publishing restrictions for every version and in every language for an item:

1. In the Content Editor, in the content tree, navigate to the item that you want to set up publishing restrictions for.
2. Click the Publish tab and in the Restrictions group, click Change.



3. In the Publishing Settings dialog box, click the Item tab.



4. To specify all the item versions as publishable, select the Publishable check box. To specify all the item versions as not publishable, clear the Publishable check box.

Note

If you clear the Publishable check box, this affects all the item versions listed on the Versions tab. This means that if you specify an item as not publishable, you cannot make a version of that item publishable. Upon publishing, the item and all its versions are removed from the target database.

5. To specify a period for when all the item versions are publishable, enter the dates and times in the Publishable From and the Publishable To fields. You can choose to specify only the start date, the end date, or leave all the fields blank, in which case all the items are publishable from the current day until you change it.

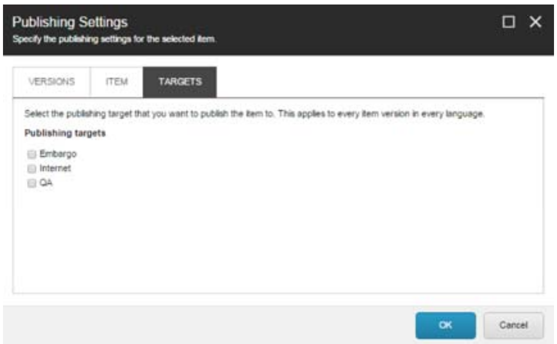
   Important

   If you specify a date range for when an item or item version is publishable, it does not mean that the item is published on the start date and removed again on the end date. Instead, it means that the item is available for publishing and included when you publish that item or the entire website. To make an item appear on your website on the start date and removed again on the end date, you must run the Publish Item wizard on both days.

6. To restrict all the item versions to be published only to certain publishing targets, select the appropriate publishing targets on the Targets tab. This applies to all versions of the item and all languages.

   If you select one or more publishing targets for the item, the item is removed from the publishing targets that you do not select when you publish.

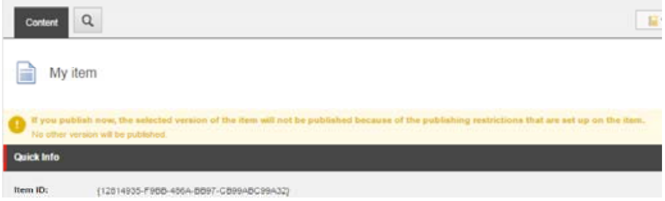
   If you do not specify any publishing target for the item, the item can be published to any publishing target you choose in the Publish wizard.



   Note

   The list of publishing targets contains both regular publishing targets and preview publishing targets.

7. When you have finished setting up the publishing restrictions, click OK. In the Content Editor, a warning appears on the Content tab of the item that informs you or other content editors that restrictions are set up for this item.



After you have set up the publishing restrictions for your items, you can preview your website in the Experience Editor to make sure that you have set up everything correctly. In the Experience Editor, you can browse through the pages on your website and see how they look on upcoming dates or in different languages.

Send feedback about the documentation to docsite@sitecore.net.

Walkthrough: Publishing items to a preview publishing target

By default, items that are in a workflow can only be published if the items are in the final workflow state. However, in situations where you need to publish an item before it reaches the final workflow state, for example, if you want the item to be reviewed by a non-Sitecore user, you can publish the item to a preview publishing target.

A preview publishing target is a publishing target that you can assign to one or more workflow states to allow items in that workflow state to be published. When you publish to a preview publishing target, all items that are in a workflow state with this preview publishing target assigned, and all items that are in a final workflow state of any workflow, are published.

In this way, items that are not in the final workflow state (and therefore cannot be published) can be published to an internal site for review or they can be published to a secure site with early release material that only certain partners or customers can access.

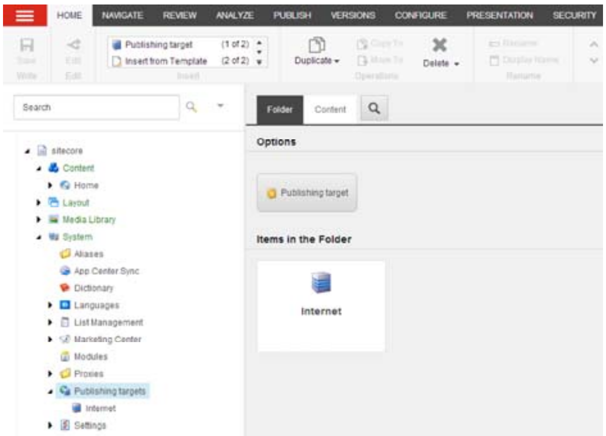
This walkthrough outlines how to:

- Create a preview publishing target
- Publish items in a workflow state to a preview publishing target

Create a preview publishing target

To create a preview publishing target:

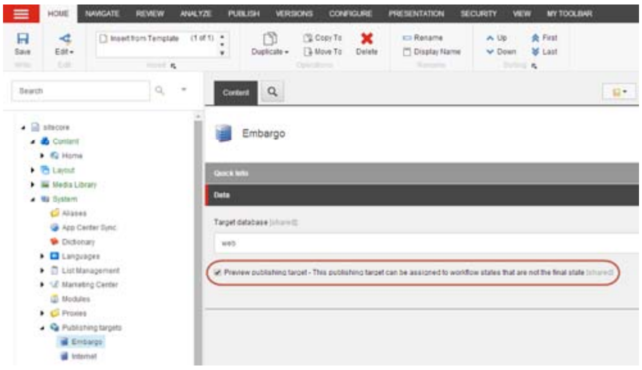
1. In the Content Editor, in the content tree, click *System/Publishing Targets*.
2. Create a new publishing target, on the Folder tab, in the Options section, click Publishing target.



3. In the system dialog box that appears, enter the name of the preview publishing target and click OK.
4. On the Content tab for the new publishing target, enter the name of your target database and select the Preview publishing target check box.

   Important

   Make sure that the name of the target database is entered correctly to avoid getting an error when you try to publish to the preview publishing target.

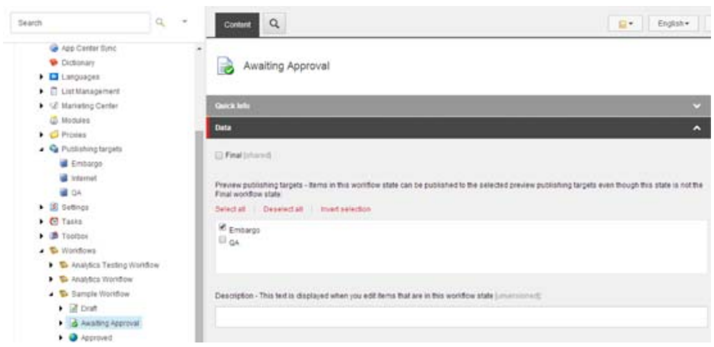


5. Click Save.

Publish items in a workflow state to a preview publishing target

To allow items in a workflow state to be published to a preview publishing target:

1. In the Content Editor, in the content tree, click *System/Workflows*, and then select the workflow state that you want to assign a preview publishing target to.
2. In the Data section, select one or more preview publishing targets to assign to this workflow state. The list contains only the publishing targets that are specified as preview publishing targets.



3. Click Save.

   Now the items that are in a workflow state with the preview publishing target assigned appear with a notification to let you know which preview publishing targets the specific item can be published to.

Note

When you publish to a preview publishing target, any publishing restrictions that are set up on the items are still valid.

Send feedback about the documentation to docsite@sitecore.net.

Build a search query

Because your Sitecore website can contain thousands of items, it is important to know how to make accurate searches and limit your search results to the most relevant items.

In Sitecore, you can run a simple search or build complex search queries. You can search using a single search query, or you can combine multiple queries to limit your search results. You can enter as many search filters and keywords as you need.

This topic outlines how to:

- Search with the default 'or' logic
- Search with predefined search options
- Search with predefined search filters
- Search with wildcards and replacements
- Search within a range
- Combine ‘and’ and ‘not’ in a search query

Search with the default 'or' logic

When you enter more than one search filter or keyword, the search uses the ‘or’ logic by default. This means that the search returns items that are based on one *or* more of the other filters or keywords.

For example, if you apply a filter for version 2 and a filter for the Japanese language in the search query, your search results will return items that have a version 2 or items that are in Japanese (or both).

Search with predefined search options

To ease the everyday work in Sitecore, the search functionality collects information on your recent search activity and the recently created or modified items. If you use these predefined search options, you can quickly get to the items that you regularly work on in the Content Editor, regardless of which item is selected in the content tree.

The predefined search options include:

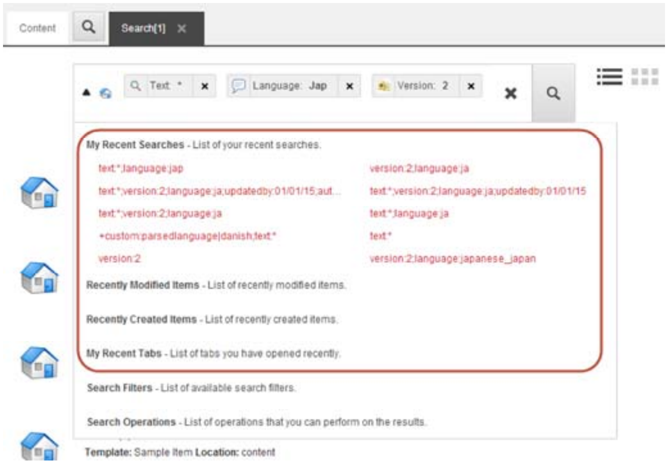
- Your recent searches
- Recently modified items
- Recently created items
- Your recently opened tabs

Note

Your Sitecore administrator can customize the default options that are available in the drop-down menu.

To open items from a predefined search option:

1. Click the drop-down arrow on the left side of the Search field.
2. Expand one of the search options to see a detailed list of the items that are listed for that specific search option.



3. Click the link to an item to open it in a new tab in the editing pane.

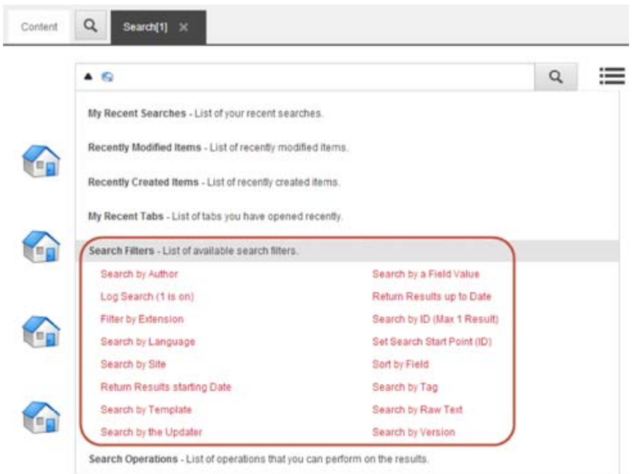
   If you click a search query under My Recent Searches, the query is inserted in the search field and you can run the search by clicking the search icon. You can enter more than one search query from My Recent Searches.

Search with predefined search filters

To limit the search results, you can insert one or more predefined search filters into the search field.

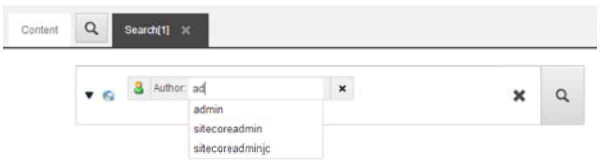
To use search filters:

1. Click the drop-down arrow on the left of the Search field.
2. Expand the Search Filters option and select the required filter.



   The filter is inserted into the Search field followed by a colon.

3. After the colon, enter the search keyword for the filter. Sitecore auto-suggests the text or prompts you to enter a date. You must enter 1-3 characters before Sitecore makes any suggestions.



4. Press ENTER or click Search .

Note

Once you get familiar with the syntax for the different searches, you can type the search filter directly into the Search field, for example *author:*, *template:*, or *start:*. Sitecore auto-suggests the text or prompts you to enter a date.

Search with wildcards and replacements

To help you search in Sitecore, you can use wildcards and replacements. Sitecore supports the following symbols and you can use them in a basic search or within a predefined search filter as needed:

- Quotation marks, for example, “text” – searches for an exact word or phrase.
- The * symbol – finds every form of a word. You can use the symbol inside a word, at the beginning or end of a word, or a combination of this. For example:
  - Searching for econom* will find "economy", "economics", “economical", and so on.

- Searching for *graph* will find “paragraph”, “paragraphing”, “graphic”, and so on.
- Using * by itself brings up all items within the item you have selected.
- The ? symbol – replaces any single character, either inside the word or at the beginning or end of the word. For example:
  - Searching for "t?re" will find "tire", "tyre", "tore", and so on.

Search within a range

If you need to search within a range, Sitecore supports both numerical and text range searches.

For example:

- price:[400 TO 500]
- title:[algeria TO bahrain]

Note

A developer can add a slide control with a predefined range, which makes it easier for you to see and search within the accepted range.

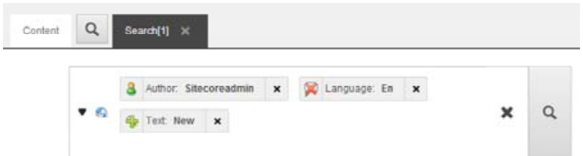
Combine ‘and’ and ‘not’ in a search query

To perform complex searches for items that, for example, must contain or must exclude specific words, you must use ‘and’, ‘or’, and ‘not’ to combine the search filters in the query.

When you enter a filter in the search field, you can click the icon for the search filter and toggle between *must* contain and *must not* contain :

- If you select , the items that the search returns must contain the value in the search filter.
- If you select , the items that the search returns must not contain the filter value.
- If your query contains both search filters that are marked with either or and search filters that are not marked, the search filter with no marking determines which items are listed at the top of the search results.

For example, in the following screenshot, the search results must have the text ‘new’ in the content, the results cannot be in the English language, and the items written by the sitecoreadmin user are listed at the top of the search results.



Send feedback about the documentation to docsite@sitecore.net.

Search for an item

In Sitecore, you can search for an item in a number of ways:

- From the Content Editor:
  - In the content area on a Search tab, where the search runs through the selected item only.
  - In the content tree, where the search runs through all the items in the content tree, starting from the selected item.
- From the Experience Editor:
  - In the Navigate to Item dialog box, where the search runs through all the items in your Sitecore installation.

This topic outlines how to:

- Search for an item in the Content Editor or the Experience Editor
- Search for an item from the content tree in the Content Editor
- Add a criterion to a search from the content tree in the Content Editor

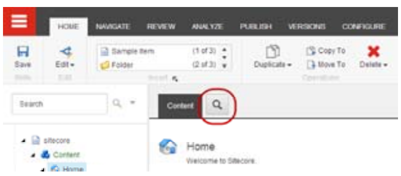
Search for an item in the Content Editor or the Experience Editor

Important

When you run a search in the Content Editor on a Search tab, you only search through the descendants of the selected item. In the Experience Editor, the search runs through *all* the items across all your Sitecore websites.

To search for an item:

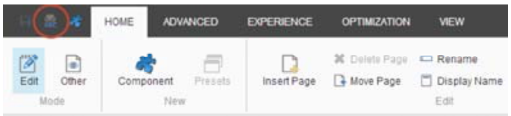
1. Open the Content Editor or the Experience Editor:
   - In the Content Editor, in the content tree, navigate to the item that you want to run your search from. In the editing pane, click Search to open a new search tab.

   Note

   If you select an item bucket, the Search tab opens by default.

- In the Experience Editor, on the ribbon, click the search icon to open the Navigate to Item dialog box, where you can search among all the Sitecore items.



2. In the Search field, enter the keywords that characterize the items that you want to search for, or build a query.

   Tip

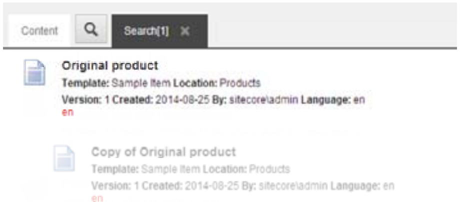
   In the Search field, you can enter * to search for all available items — in the Content Editor, this means all the descendants of the selected item and in the Experience Editor, this means all items in your Sitecore installation.

3. Press Enter or click Search .
4. To filter the search results, use the facets to the right of the search results, and to change the view of the results, you can click one of the view buttons to the right of the Search field.

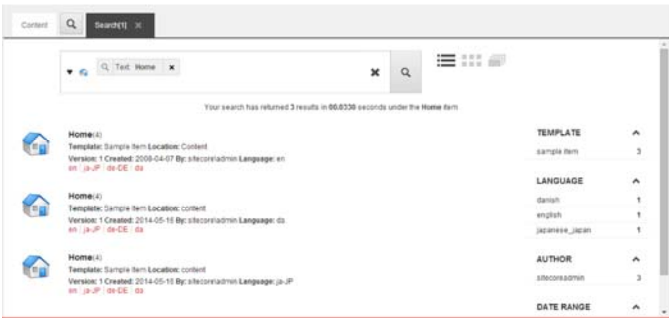
   Note

   If you change the view, the search is launched again based only on the keywords and filters in the Search field.

   Note that in the search results, a clone is greyed-out and represented with an indent.



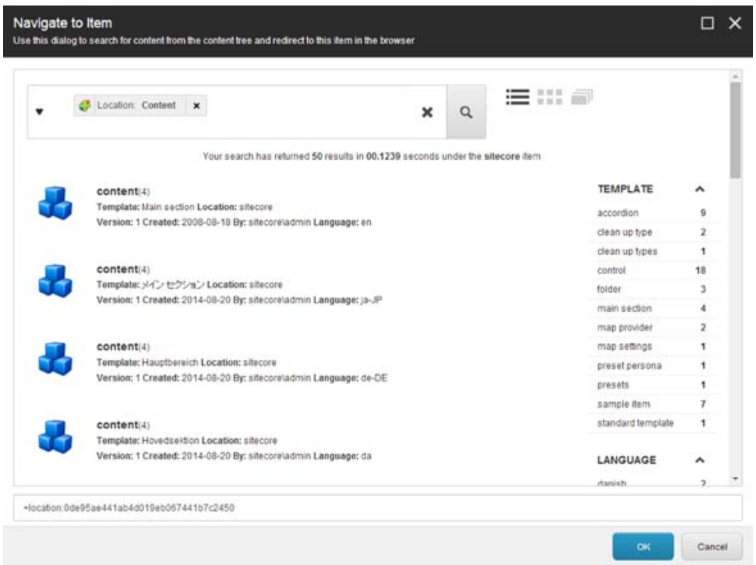
5. To open an item from the search results:
   - In the Content Editor, click the title of the item to open it in a new tab.



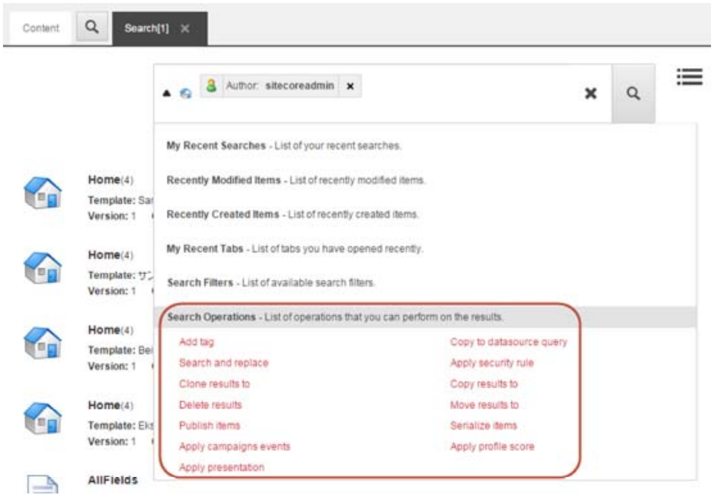
   - In the Experience Editor, select the item and click OK. The Experience Editor then redirects to the page where the selected item is located.

     Tip

     If you are working on multiple websites, you can search for an item on another website and use this search functionality to redirect to the Experience Editor from one site to another.



6. Alternatively, you can perform certain operations on all the items directly from the search. To [perform an operation on all the search result items](), click the down arrow to the left of the search field, expand Search Operations, and select the operation that you want to perform on all your search results.



Search for an item from the content tree in the Content Editor

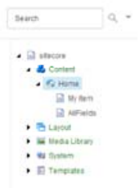
If you want to do a quick and simple search through the content tree, you can use the search function associated with the content tree in the Content Editor.

Note

The search runs through the entire content tree starting from the selected item. This means that the search results found at the starting point are listed first.

To search for an item using the content tree search:

1. In the content tree, select the item that you want to start the search from.
2. In the Search field, enter the keyword that you want to search for. This can be any character, number, word, and so on.



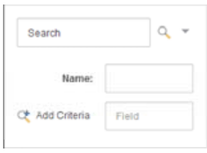
3. Click Search or press ENTER to perform the search.

Add a criterion to a search from the content tree in the Content Editor

If you need to search in a specific field or for more than just one item or keyword, you can, from the drop-down menu, perform a search in a specific field and add more criteria to the search.

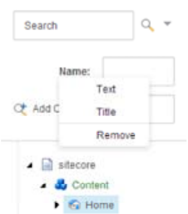
To add a criterion to your search:

1. In the content tree, select the item that you want to start the search from.
2. In the Search field, click the drop-down arrow.

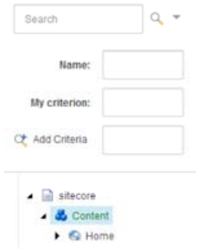


3. To perform a search only in the items' Name field, in the Name field, enter the name of the item you want to search for.

   If you want to search in a different field, click Name, and in the drop-down menu that appears, all the fields in the current item are displayed. Click the field that you want to search in and enter the relevant search word.



4. To add a custom criterion, in the Add Criteria field, enter the name of the criterion and then click Add Criteria.



You can click the name of the criterion to change it to one of the predefined criteria.

5. Enter the value for the custom criterion and press ENTER. To remove the criterion, click the criterion name and then click Remove at the bottom of the menu that appears.
6. To open a search result in the content area, click a search result.

Send feedback about the documentation to docsite@sitecore.net.

Searching

The search engine in Sitecore lets you quickly search through all your items in the content tree. To help you locate exactly the item you are looking for, you can use the advanced search functionality, such as search filters and facets.

Note

The search does not include your archived items.

You can access the Sitecore search engine from every item in the content tree in the Content Editor or from the ribbon in the Experience Editor.

The Content Editor

In the Content Editor, there is a Search tab next to the content tab of every item. Item buckets open directly on the Search tab because you need the search functionality to find and open hidden items in an item bucket.

To run a search on an item in the content tree, in the content area, click Search and, in the new Search tab, enter your search criteria. The search runs through the selected item only.

cms_ce_ContentArea_SearchTab

If you want to run more than one search from the same item, you just open a new Search tab. You can open as many search tabs as you want. Each Search tab has a time stamp that gives you a reference to each specific search, for example, if you want to save a search.

The Experience Editor

To open the search in the Experience Editor, in the ribbon, click Search . The Navigate to Item dialog box opens and runs by default a search for all the items in Sitecore.

cms_xe_NavigateToItem_Dialog

The search interface

Whether you access a search from the Content Editor or the Experience Editor, you have the same options to build a query, to view, and to filter the search results.

cms_all_SearchInterface

Search field

In the Search field, you enter the search query for your search. You can enter either free text, use the predefined search filters, or a combination of both. You can access the search options from the drop-down menu.

More search options

If you click the drop-down arrow to the left of the search field, you open a list of pre-defined search options that you can use to optimize and speed up your search. You can, for example, open an item from a list of the most recently modified or opened items, select one of the pre-defined search filters, or apply one of the search operations to all the items returned by the search.

Note

The administrator sets up the search engine and determines, for example, which search views and facets are available to you and whether or not you are able to see media items in the search results.

View types

The different views of the search results are available to the right of the search field. The default views are *List View*, *Grid View*, and *Image View* . If necessary, your administrator can activate several other views such as Tag View or Table View.

Note

If you change the view, the search is launched again, based only on the keywords and filters in the Search field.

Facets

A facet is a way of categorizing the items returned by a search. For example, the items can be classified by author, language, or by when they were created. Once you have a list of search results, you can use the facets to filter the results further.

Search results

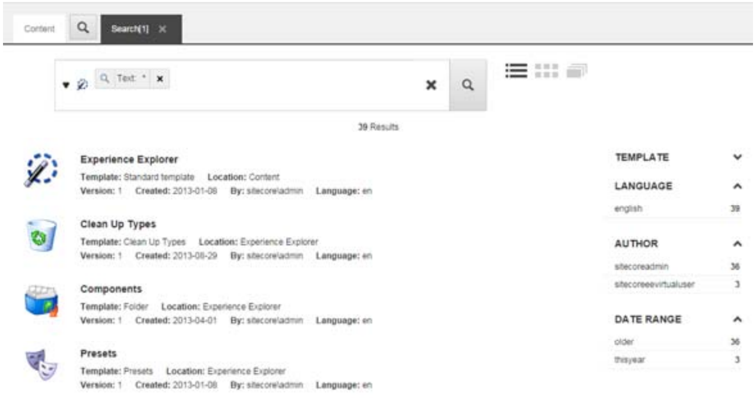
When you run a search, the results are listed below the search field. The search results contain basic information about the items returned, such as title, template, and version to give you a quick overview. From the list of search results, you can click an item to open it in the Content Editor or to navigate to the appropriate page in the Experience Editor.

Send feedback about the documentation to docsite@sitecore.net.

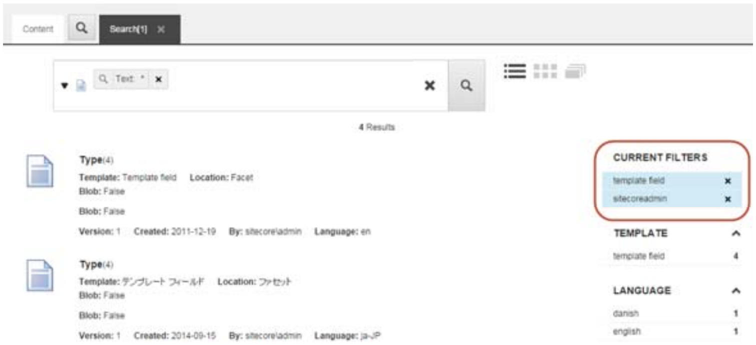
Filter the search results

When you perform a search in Sitecore and it returns too many results, you can quickly refine the search by filtering on the facets that are available on the right of the search pane.

A facet is a way of categorizing items. For example, you can categorize your items by author, language, or a date range.



To filter your search results, click a facet and then the relevant filter. When you click a filer, it is highlighted and listed under Current Filters.



To clear a filter, under Current Filters, click the filter that you want to remove.

Send feedback about the documentation to docsite@sitecore.net.

Perform an operation on search results

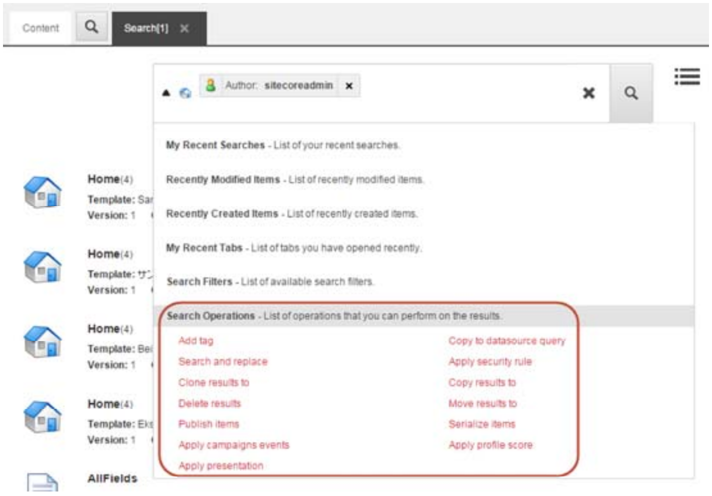
Search operations is a powerful feature that lets you run a number of operations on all the items in the search results. For example, you can use the search operation feature to add a tag or apply the same security rules to multiple items at the same time.

Note

The operations in the following screenshot are the default search operations in Sitecore, but your administrator can create and add customized search operations.

To apply a search operation on all the items in the search results:

1. In your list of search results, click the down arrow to the left of the Search field.
2. Expand Search Operations and select the [search operation](#) that you want to apply to all the items in the search result.



Note

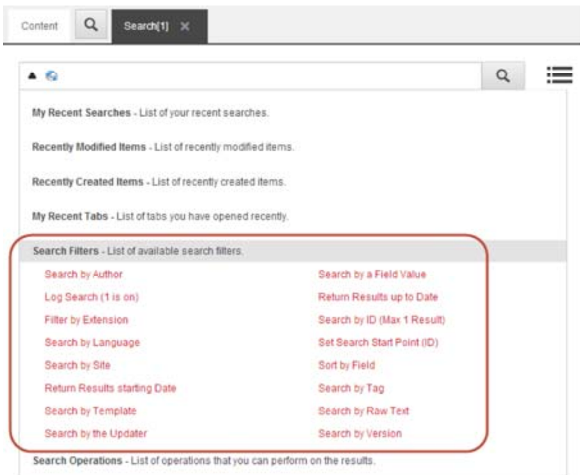
The search operations available to you in the menu depend on your access rights.

3. Depending on the search operation you click, you must complete the operation by following the steps in the dialog box that opens.

Send feedback about the documentation to docsite@sitecore.net.

The search filters

When you build a search query in Sitecore, you can select the following search filters from the drop-down menu in the Search field.



When you select a filter, it is inserted into the Search field and your search results will be filtered accordingly.

Search filter	Description
Search by author	Search for the items created by a specific author.
Log search (1 is on)	Turn on logging for a single search query (1 is on, 0 is off.) Only developers and administrators should use this filter.
Filter by extension	Search for items with a specific extension. This is useful when searching for files in the Media Library.
Search by language	Search for items in a specific language.
Search by site	Search for items on a specific site. This is useful when you have multiple sites on the same Sitecore installation.
Return results starting date	Search for items that were created on or after a specific date.
Search by template	Search for items based on a specific template.

Search by version	Search for items with a specific version number.
Search by a field value	Search for items with a specific value in a specific field. For example, to search for all the items that have the word design in the Title field, enter *custom:title|design*.
Return results up to date	Search for items that were created on or before a specific date.
Search by ID (max 1 result)	Search for an item with a specific Item ID. You must enter the whole ID.
Set search start point (ID)	Search for items from a specific location. Enter the specific content item ID where you want to start the search.
Sort by field	Sort the items based on the text in the field specified after the colon. For example, to sort the search results by item name you enter *sort:_name*.
Search by tag	Search for items that have a specific tag.
Search by raw text	Search for items that contain a specific word.

Send feedback about the documentation to docsite@sitecore.net.

The search operations

When you search for items from the Content Editor or from the Experience Editor, you can use the search operations to perform actions on the search results. Directly from the Search window, you can perform the operations as described in the following table.

Search operation	Description
Add tag	Tag all the items in the search results with the same tag. To add tags to the items in the search result, click Add tag and the Insert Item dialog box opens. Find the tag you want to add to items in the search result, either from the Items option, where you can navigate to the tag in the content tree, or from the Search option, where you can use the search interface to find the tag.
Search and replace	Replace a specific word in all the search result items with another word. When you click Search and replace a dialog box appears. Enter the word you would like to replace, then enter a pipe '|' followed by the word you want to insert instead. For example, to replace the words 'Office Core' with 'Sitecore', enter *Office Core|Sitecore* in the text field. Note The search and replace functionality is case-sensitive. If you need to replace multiple words, such as 'Office Core', they are not replaced when they are joined by, for example, a hyphen (Office-Core).
Clone results to	Clone all the items in the search results to a specific location in the content tree. When you click Clone results to, the Clone Item dialog box opens. Here you specify the location for the new clones. You can either browse to the location through the content tree on the Browse tab or click the Search tab to search for the location.
Delete results	Permanently delete all the items that appear in the search result. When you click Delete results, all the search result items are instantly deleted from Sitecore and a message appears telling you that the items have been deleted. Warning You are not presented with a warning or asked to confirm your choice before the items are deleted. Once you click Delete Results, all the search results are completely deleted from Sitecore and you are not able to restore them from the Recycle Bin.
Publish items	Publish the items in the search result at the same time. When the publishing job is finished, a message appears telling you that the items have been published.
Apply campaign events	Apply campaigns, events, or failure actions to the items in the search result. When you click Apply Campaign Events the Attributes window opens where you can specify the campaigns, events, failure actions, and settings for all the items in the search result.
Apply presentation	When you click Apply Presentation the Layout Details window opens. From here you can assign the same layout, controls, and placeholders to all the items in the search result.
Copy to datasource query	Create a text string that you can then paste into a "data source" field, so that the data source returns the items that match this search result. This operation is only relevant for developers and administrators.

Apply security rule	This operation should only be performed by security administrators. It allows them to change the security settings for the content items listed in the search results. When you click Apply security rule the Security Settings windows opens where you can assign access rights to all the items in the search result for either a specific user or a specific role.
Copy results to	Copy all the items in the search result to another folder. When you click Copy results to, the Copy Item To dialog box opens where you can browse to or search for the folder that you want to copy the items to.
Move results to	Move all the items in the search result to another folder. When you click Move results to, the Move Item To dialog box opens where you can browse to or search for the folder that you want to move the items to.
Serialize items	Serialize all the items in the search results to files in the standard serialized files folder. When serializing an item, it is converted into a format that can be written into a file and imported again later into this or another Sitecore installation. When the operation is finished, a message appears telling you that the items have been serialized.
Apply profile score	Assign a profile card or a custom profile score to the items listed in the search result. When you click Apply profile score the Profile cards window opens where you can select a profile card or set a customized profile score for the items in the search result.

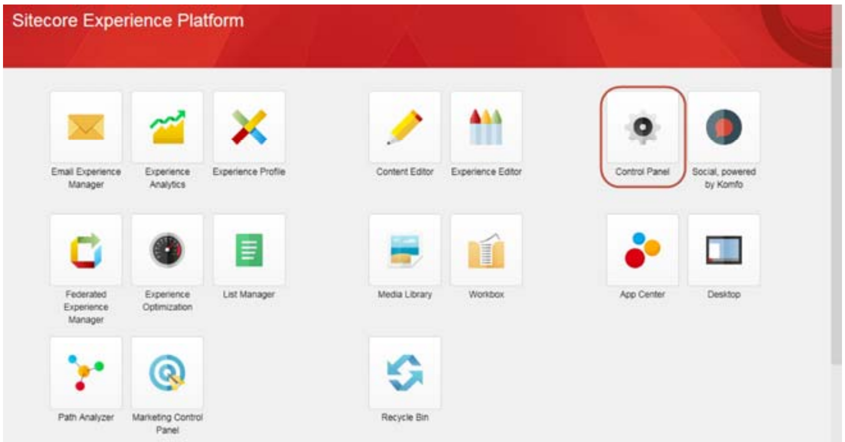
Send feedback about the documentation to docsite@sitecore.net.

Change your personal settings

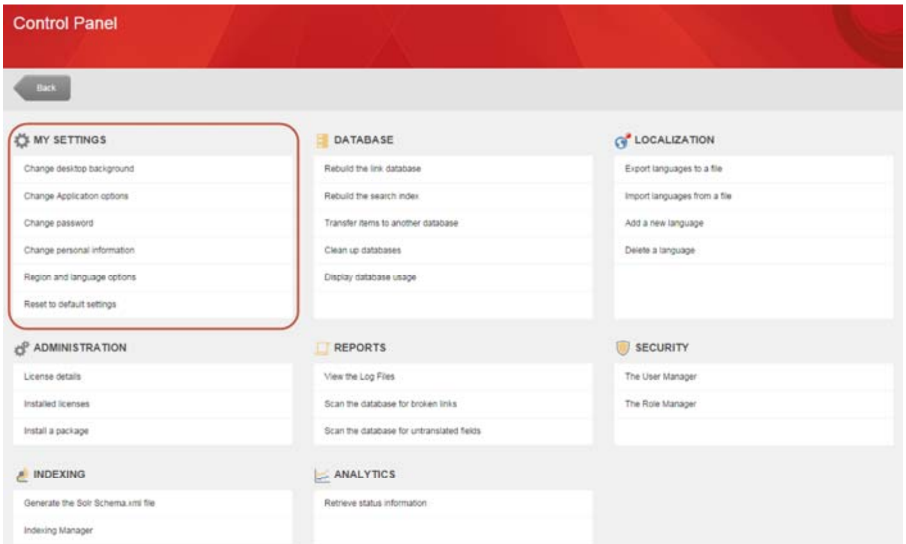
In the Sitecore Control Panel, you can set up your personal settings, for example your password or the region and language options, to make the application suit your needs.

To change your personal settings:

1. On the Sitecore Launchpad, click the Control Panel.



2. In the Control Panel, in the My settings section, you can change the following settings:



- Change desktop background – in the Background field, select the image that you want to use, specify the position of the image and the color that you want to surround the image if it does not fill the screen.

- Change Application options – in the Application Options dialog box, on the Content Editor tab, specify which elements you want to display on the Content Editor. On the View tab, you can customize the content tree.
- Change password – in the Change Password dialog box, enter your current password and then enter and confirm your new password.
- Change personal information – in the Personal Information dialog box, you can change your name, email address, and the image that you use as your portrait throughout Sitecore.
- Region and language options – in the Regional and Language Settings dialog box, you can specify the formats that you want to use for numbers, currencies, times, and dates. In addition, you can specify the language that you want the client application to use.
- Reset to default settings – click to reset all your personal settings to the default settings.

3. When you have finished setting up your personal information, click Back to return to the Launchpad.

Send feedback about the documentation to docsite@sitecore.net.

The editing tools

In Sitecore, there are two editing tools where you can create and edit the content on your website:

- The Content Editor – an application that is designed for more experienced content authors who are familiar with Sitecore and the functionality that it contains.
- The Experience Editor – an intuitive editing environment that is designed for content editors who want to edit and write content directly on the page.

In the Content Editor and the Experience Editor, the items that make up your website are presented in very different ways. In the Content Editor, the items are objects in the content tree and when you select an item, you can edit its fields. In the Experience Editor, the items are presented as they look on the website and you can edit them directly on the page.

For example, in the Content Editor, the Services item of a sample website looks like this:

cms_ce_ContentTab

When you click the Services item in the content tree, the Content tab displays the Services item's field values, which you can also edit. If you want to edit another item, you must click the item in the content tree.

In the Experience Editor, you see the Services item presented as it looks on your website:

cms_xe_HomeTab

You can edit all of the items on the page – text, graphics, logos, links, and so on, even if they do not belong to the selected item.

Send feedback about the documentation to docsite@sitecore.net.

Customize the Content Editor

The Content Editor contains a number of user-interface elements that you can customize:

- The Content Editor elements – you can decide which elements of the Content Editor to display.
- The Content Editor ribbon – you can hide tabs, hide the entire ribbon, or create your own toolbar within the ribbon.
- The content tree – you can add quick actions, resize the content tree pane, or hide it entirely.

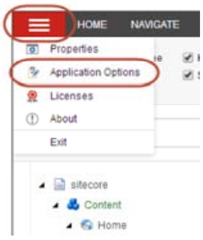
This topic outlines how to:

- Customize elements in the Content Editor
- Customize the Content Editor ribbon
- Customize the content tree

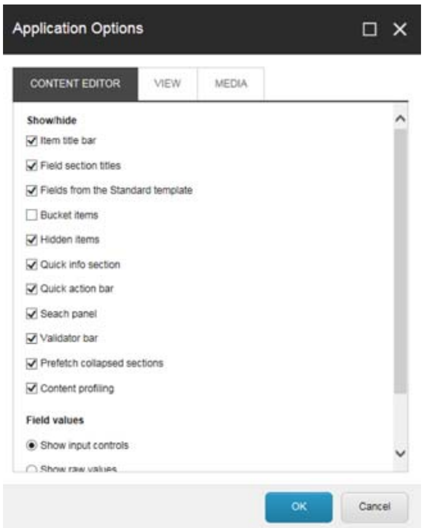
Customize elements in the Content Editor

To set up the application options for the Content Editor:

1. In the Content Editor, click the Content Editor menu, and then click Application Options.



2. In the Application Options dialog box:
   - On the Content Editor tab, select or clear any of the check boxes to hide or show the relevant element.
   - On the View tab, specify how you want the content tree to be displayed.
   - On the Media tab, specify the display options for media files.

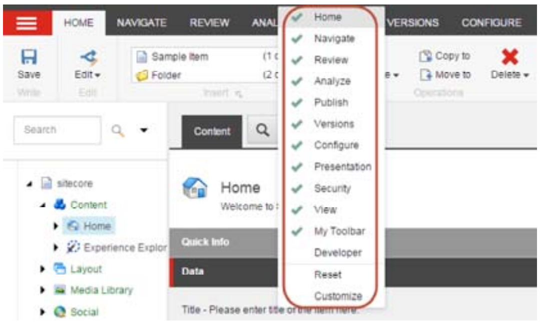


3. When you have finished configuring the application options, click OK.

Customize the Content Editor ribbon

To customize the Content Editor ribbon:

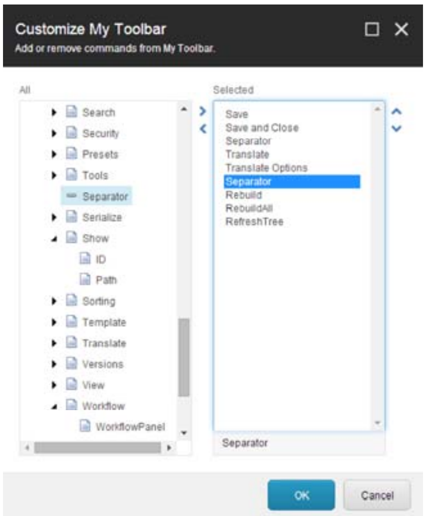
1. In the Content Editor, right-click anywhere on the ribbon.
2. In the menu that appears, clear the check boxes of the tabs that you want to hide in the ribbon. To restore the default tab selection, click Reset.



Note

To hide the ribbon entirely and only display the tabs, double-click any tab on the Content Editor ribbon. To display the ribbon again, click any tab.

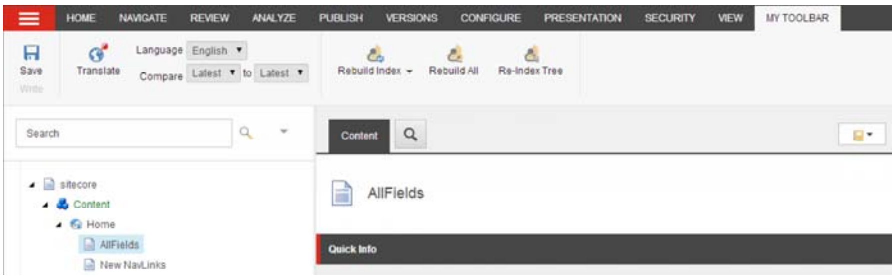
3. To create your own toolbar in the ribbon, click Customize.
4. In the Customize My Toolbar dialog box, in the All pane, expand the groups and select the commands that you want on your toolbar.



Note

You can only add commands and not the groups to your toolbar. However, you can add the command Separator to divide the added commands into groups. Use the up and down arrows to place the commands in the order that you want.

5. When you have finished, click OK.
6. Right-click on the Content Editor ribbon and, in the menu, select My Toolbar. The My Toolbar tab is added to the ribbon and contains all the commands that you just selected.



Customize the content tree

In the Content Editor, you can choose to hide or show certain elements in the content tree.

Hide elements

On the View tab:

- To hide the content tree, clear the Content Tree check box.
- To hide the root of the content tree (sitecore) and the Media Library, clear the Entire Tree check box. In this way, users can only see the items that are part of their website.
- To hide some of the system items that are displayed in the content tree, clear the Hidden Items check box.

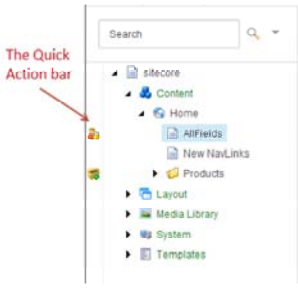
The other two settings, Raw Values and Standard Fields, are only relevant for administrators

Note

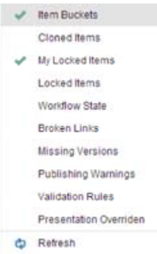
If the content tree takes up too much space, you can drag the bar between the content tree and the editing pane left or right, to resize the content tree pane. In addition, you can press F11 to expand the browser window to fill the entire screen and hide the user-interface elements, such as the title bar, menus, toolbar, and the status bar.

Display quick actions

The Quick Action Bar runs down the side of the content tree and can display useful information about the items in the content tree. For example, the Quick Action Bar can display quick actions that tell you which items are item buckets, cloned items, or locked items.



When you right-click the Quick Action Bar, a floating menu appears with a list of the available quick actions that you can click to display or hide next to the items in the content tree. You can display as many of the quick actions as you like, but the performance might decrease for every action you select.



Note

The quick actions that are available to you can vary. This is controlled by your Sitecore system administrator.

Some of the quick actions are only for information while others you can click on to perform an action on the item.

The quick actions are:

- Item Buckets – appears next to the items that are item buckets.
- Clones Items – appears next to the items that are a clone of another item. Click the icon to open the original item.
- My Locked items – appears next to the items that you have locked. Click the icon to unlock the item.
- Locked items – appears next to the items that are locked by other users. Place your mouse over the icon and the tooltip shows you who locked the item.
- Workflow state – appears next to the items that are in a workflow state. If you have the necessary permissions, you can click the icon and move the item to the next workflow state.
- Broken Links – appears next to items that contain broken links.
- Missing Version – appears next to the items that are missing a language version.
- Published Warnings – appears next to the items that have publishing warnings.
- Validation Rules – appears next to the items that contain validation errors.
- Presentation Overridden – appears next to the items where the template layout settings are overridden by another layout.

Important

The Validation Rules and Broken Links quick actions slow down performance the most. Every time you open an item in the content tree, Sitecore must refresh these quick actions.

Send feedback about the documentation to docsite@sitecore.net.

Set up an email reminder

In Sitecore, you can schedule to send an email reminder to one or more users. This feature is particularly useful if you have time-sensitive content on your website. For example, if your website contains a special offer that runs out on a certain date, you can use the reminder to send yourself or others a reminder to remove the offer from the website on the appropriate day.

To set a reminder:

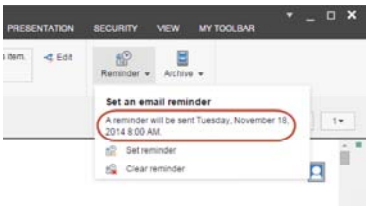
1. In the Content Editor, on the Review tab, in the Schedule group, click Reminder. In the list of options, select Set Reminder.

2. In the Set Reminder Wizard, on the Welcome to the Set Reminder Wizard page, click Next.
3. In the Reminder Text dialog box, in the Reminder text field, enter the text that you want to be sent to recipients as a reminder, and then click Next.

4. In the Date and Recipients dialog box, in the Date fields, enter the date and time when you want the email reminder to be sent. In the Recipients field, enter the email addresses of the recipients, and then click Set.

5. Click Finish.

To verify the sending date of the reminder, on the Review tab, in the Schedule group, click Reminder. The reminder details are displayed above the Set Reminder option.



To clear the reminder that you have configured, on the Review tab, in the Schedule group, click Reminder, and then click Clear Reminder.

Note

You can only configure one reminder for each item. If you already have one reminder set for an item, and want to set another reminder, you overwrite the earlier reminder.

Send feedback about the documentation to docsite@sitecore.net.

Subscribe to a client RSS feed

To stay informed about updates to items or [workflows](#) in Sitecore, you can subscribe to a client RSS feed for that item.

In Sitecore, there are three types of client RSS feeds:

- Item feed – this keeps you up to date on the workflow states that the item or a version of the item goes through. This is useful for content authors to keep track of specific items.
- Workflow feed – this keeps you up to date on the entire workflow. This is useful for administrators and super users who have access to every workflow state that the workflow contains.
- Workflow state feed - this client feed keeps you up to date on this particular workflow state. This is useful for content authors and managers who only have access to particular workflow states.

This topic outlines how to:

- [Subscribe to an item feed]
- [Subscribe to a workflow feed]
- [View your client RSS feeds]

Subscribe to an item feed

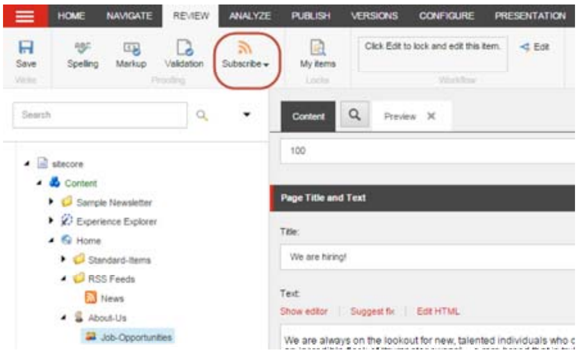
When you subscribe to an item feed, you receive updates for each time the workflow state changes for the item or a version of the item.

Note

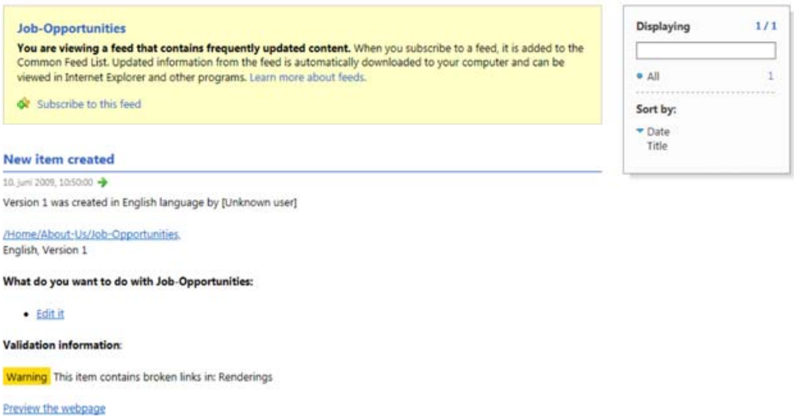
Only if the item is subject to a workflow will the RSS feeds contain useful information.

To subscribe to a client RSS feed for an item:

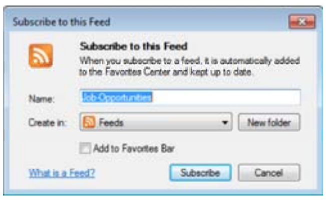
1. In the Content Editor, navigate to the item that you want to subscribe to.
2. Click the Review tab, and then in the Proofing group, click Subscribe.



3. In the webpage that opens, click Subscribe to this feed



4. In the Subscribe to this feed dialog box, enter an appropriate name and location for the feed, select the Add to Favorites Bar check box, and click OK.



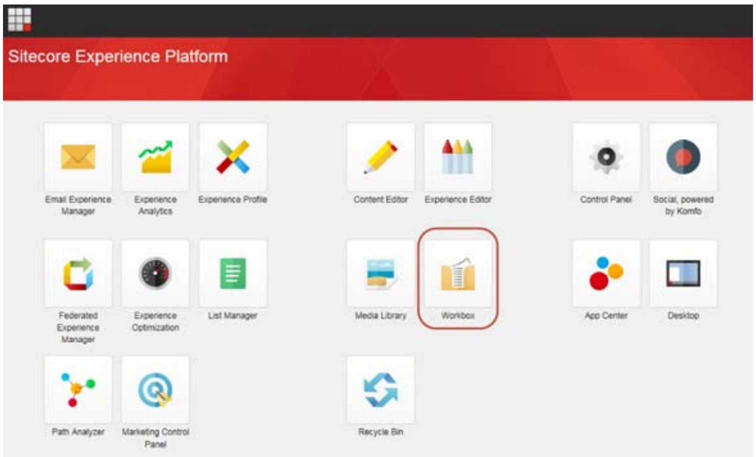
Subscribe to a workflow feed

When you subscribe to a workflow feed, you receive updates for:

- Workflow feeds every time the state changes for the item.
- Workflow state feeds every time a version of an item enters or leaves a specific workflow state.

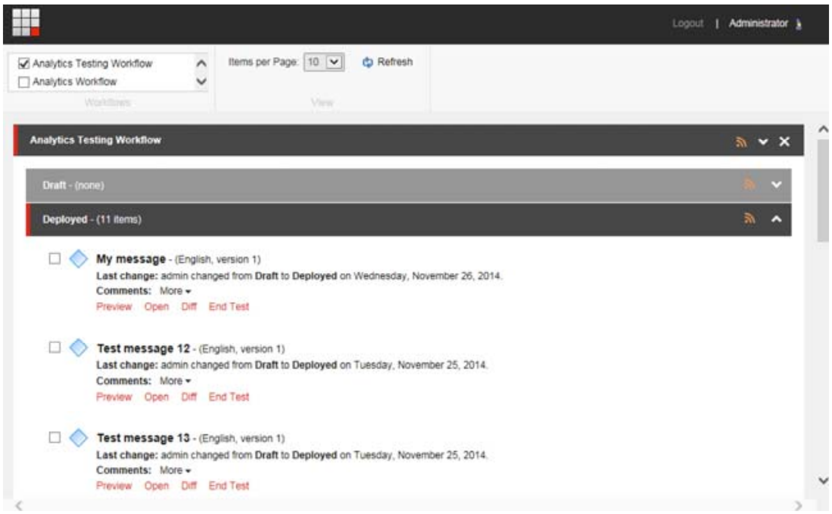
To subscribe to a workflow feed:

1. From the Launchpad, click the Workbox.



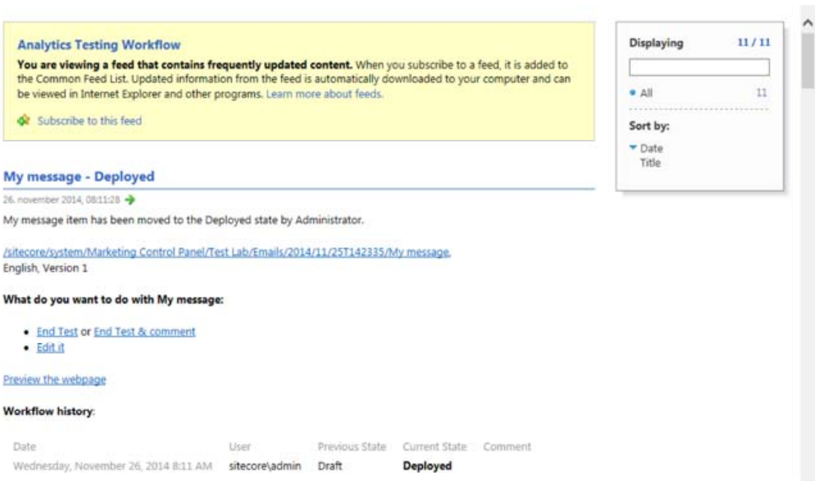
Alternatively, you can open the Workbox from the Content Editor, on the Pages bar at the bottom.

2. In the Workbox ribbon, in the Workflows group, select the workflow that you want to subscribe to:



- To subscribe to a workflow feed, click RSS feed to the right of the workflow.
- To subscribe to a workflow state feed, click RSS feed to the right of the workflow state.

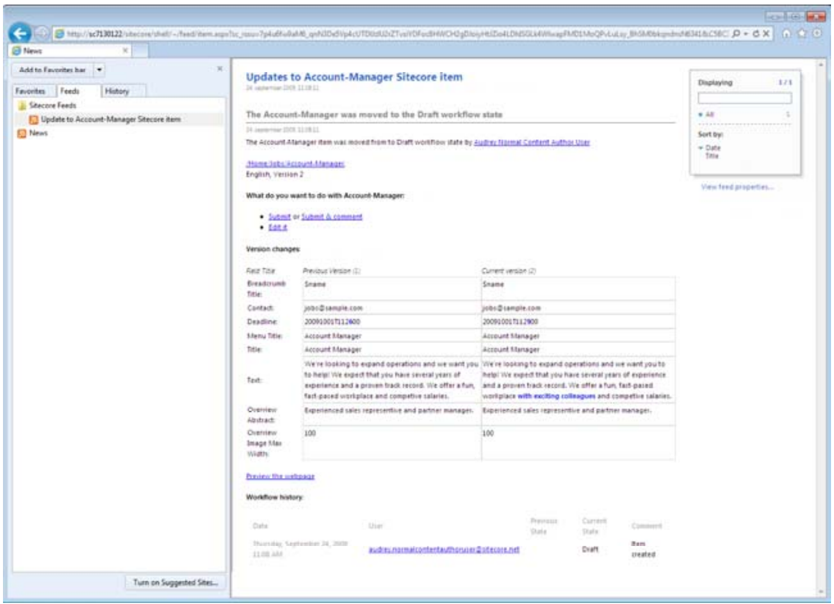
3. In the webpage that opens, click Subscribe to this feed.



4. In the Subscribe to this feed dialog box, enter an appropriate name and location for the feed, select the Add to Favorites Bar check box, and click OK.

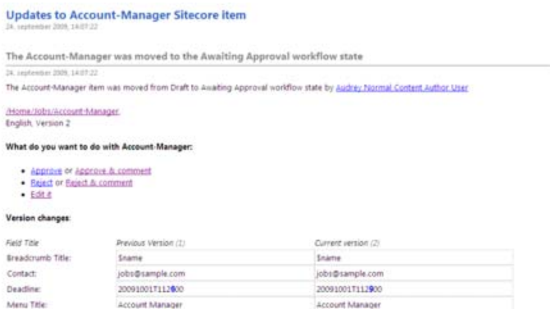
View your client RSS feeds

You can view all the client RSS feeds that you have subscribed to in your browser. In Internet Explorer, in the Favorites panel, click the Feeds tab, and then click the link to the item feed.



This webpage contains some information about the item, including:

- The name of the item.
- Its current workflow state.
- The workflow history of the item.
- A table showing the differences between the new version and the old version of the item.



In addition, the client RSS feed contains some workflow commands that let you move the item through the workflow — if you have the appropriate permissions. In this example, the user can:

- Approve or reject the item.
- Approve or reject the item and insert a comment.
- Edit the item.

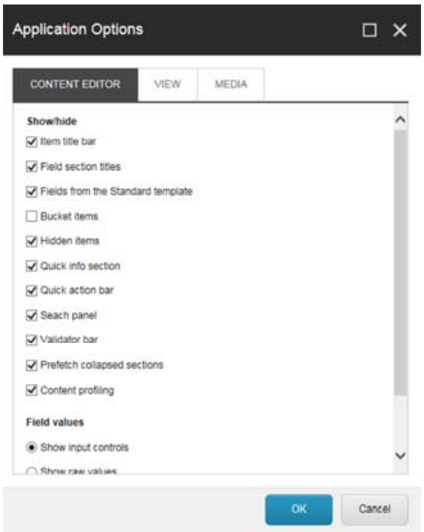
Important

If you want to perform tasks from the client RSS feed, you must enter your Sitecore login credentials to be directed to the appropriate page. However, if you select the Remember me check box on the Sitecore login page, Sitecore remembers your login and you are directed to the page automatically.

Send feedback about the documentation to docsite@sitecore.net.

The Application Options

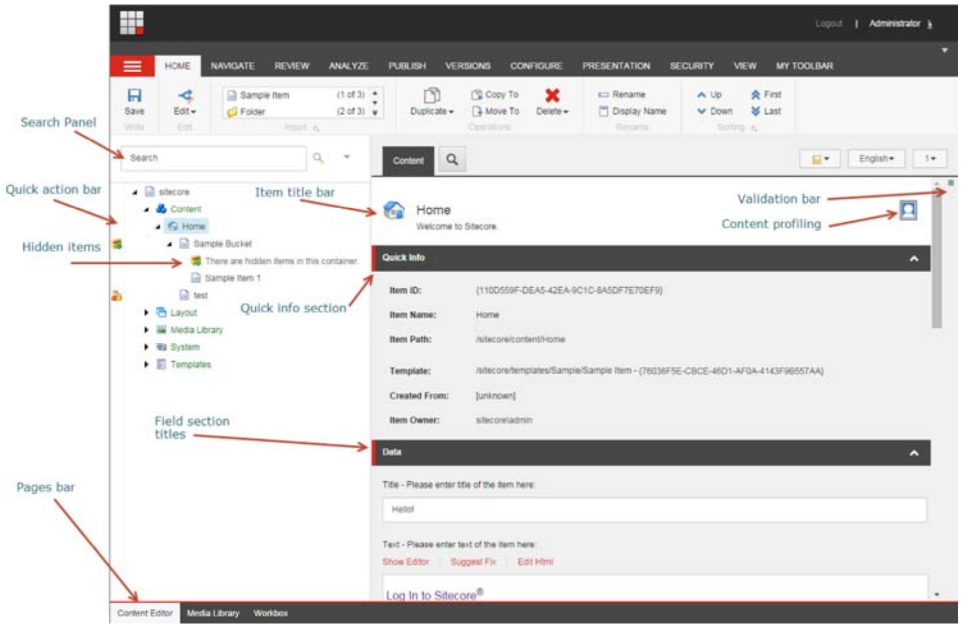
In the Application Options dialog box, you can specify which elements you want to display in the Content Editor and in the Drag & Drop dialog box when you upload media files to the Media Library.



The Content Editor tab

On the Content Editor tab, you can specify what elements you want to display in the Content Editor.

Section	Field	Description
Show/hide	Item title bar	Show or hide the Item Title bar.
	Field section titles	Show or hide the Field Section titles.
	Fields from the Standard template	Display a large number of fields that are used internally by Sitecore. These are hidden to improve performance.
	Bucket items	Display the otherwise hidden items in an item bucket in a folder structure. You should clear this check box because it will prevent the system from unnecessarily loading all the items in the content tree.
	Hidden items	Show or hide the hidden items in the content tree.
	Quick info section	Show or hide the Quick Info section.
	Quick action bar	Show or hide the Quick Action Bar.
	Search panel	Show or hide the Search panel.
	Validator bar	Show or hide the Validator bar.
	Prefetch collapsed sections	The system fetches and keeps the fields from the Field Sections in memory before the section is opened.
	Content profiling	Show or hide the Content Profiling buttons.
Field values	Show input controls	Shows the field values as input controls. This is only relevant for administrators.
	Show raw values	Shows the field values as raw values. This is only relevant for administrators.
Pages	Show pages bar	Determines whether the shortcuts to the Sitecore applications are displayed at the bottom of the Content Editor.



The View tab

On the View tab, more options determine what is displayed in the content tree and the Content Editor and the Marketing Center.

Section	Field	Description
Content tree – item names	Display name (language-specific name)	Select to show the language-specific display name that you can give an item.
	Item key	Select to show the item key in the content tree.
Content tree root	Show the content tree in the Content Editor	Show the content tree in the Content Editor
	Show entire content tree	Show the entire content tree in the Content Editor. Clear the check box to show only the Content node in the content tree.
Control properties	Show the Personalization section	To be used only in the Marketing Center.
	Show the Test Lab section	To be used only in the Marketing Center.

The Media tab

On the Media tab, you can specify what you want to see in the Drag & Drop dialog box or whether to open the editing application available on the user's computer.

Note

The drag-and-drop functionality is currently only supported in Internet Explorer 7 or later. If you use a different browser such as Mozilla Firefox, you must use the Upload files and Upload files (advanced) options to upload your files to the Media Library.

Section	Field	Description
Drag & Drop	Show only latest version of files	Select to show only the latest version of the media files in the Drag & Drop dialog box.
	Show versioned files in folders	Select to show all versions of a media file in the Drag & Drop dialog box.
Media edit button default	Open editing application installed on user's computer.	Open the media items with the editing application that is installed on the user's computer.

Send feedback about the documentation to docsite@sitecore.net.

The Content Editor

The Content Editor is an editing tool that you can use to manage and edit all the content on your website. It is designed for more experienced content authors who are familiar with Sitecore and the functionality that it contains.

The Content Editors appearance and functionality vary depending on the user's roles, the local security settings, and the customizations that have been implemented on the Sitecore installation.

The user interface

The Content Editor's user interface consists of three main areas that you can customize to fit your individual needs when you work in the Content Editor. The three areas are:

- The ribbon – the area where all the functionality is available.
- The content tree – the area where all the items are organized.
- The content area – the area where you can edit your items.

cms_ce_TheContentEditor

The ribbon

The ribbon consists of the System menu and tabs where all the functionality that you can use in the Content Editor is available.

Depending on your access rights and security settings, the functionalities that are available to you in the ribbon can vary. As a minimum, you can expect to see:

cms_ce_TheContentEditor

If you have more extensive access rights, you have access to more commands and more tabs. If you have complete access to the Content Editor or if you are a Sitecore administrator, you will see something like this:

cms_ce_FullRibbon

The content tree

One of the main features of the Content Editor is the content tree, which presents all the items that are created in your Sitecore installation.

cms_ce_ContentTree

The Sitecore content tree has a similar structure to the one used in Windows Explorer, however, the Sitecore objects and groups are not represented by standard folder and file icons but by many different icons.

You can expand and open every group in the Sitecore content tree just as you can with the folders in Windows Explorer. When there is a plus '+' beside an icon, it contains subitems and can be expanded. When there is a minus '-' beside an icon, it has already been expanded and when there is no symbol, the item contains no subitems.

The content area

The content area is where you edit an item. When you click an item in the content tree, two tabs appear in the content area:

- The Content tab, where you can edit any of the item's field values.

  Note

  If you do not see a field as expected, on the ribbon, click the View tab, and then select the Standard Fields check box. This shows all the fields from the Standard template on every item that you open.

- The Search tab, where you can [search through the selected item](#).

In the top-right corner of the content area, additional functionality is available that is specific to the item that is selected.

cms_ce_ContentAreaFunctionalities

The functionalities are:

- Navigate – navigate to a specific section or field in the list of field values.
- Language – toggle between the language versions of the item or to create a new language version.
- Versions – toggle between versions of the item or to create a new version.
- Validation warnings – move your mouse over the individual icons to see the warnings related to the current item.
- Edit Profile Cards – edit the profile cards that are associated to the item. This is mostly relevant if you are a marketing user.

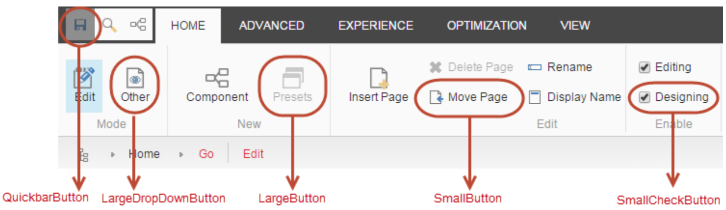
Send feedback about the documentation to docsite@sitecore.net.

Customize the Experience Editor ribbon

To customize the Experience Editor ribbon, you can create a new button item and assign a command to the button. Use the *PageCodeScriptFileName* property where you can specify a path to a JavaScript file that you want to execute when the ribbon tab loads.

You can create the following ribbon controls:

- QuickbarButton
- LargeDropDownButton
- LargeButton
- SmallButton
- SmallCheckButton



Note

For these controls to support SPEAK commands, make sure that the controls inherit the *CommandRibbonComponent* template (ID: "{CE11D144-D6A8-4B33-8E98-07DED610E952}"), which is located in */sitecore/client/Applications/ExperienceEditor/Common/Templates/Controls.*

This topic outlines how to:

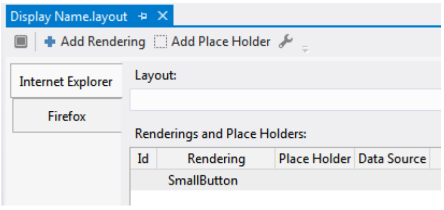
- [Create a new button item](#)
- [Hide an existing ribbon control](#)

Create a new button item

To create the new button item for the Experience Editor ribbon:

1. In the Core database, open the Content Editor and navigate to */sitecore/content/Applications/WebEdit/Ribbons/WebEdit/Page Editor/Edit*.
2. Create a new item based on the relevant ribbon control template, for example, the *Small Button* template. The templates are located at */sitecore/templates/System/Ribbon/*.

3. For the new item, add the following information:
   - In the Header field, enter the display name of the button.
   - In the ID field, enter a unique identifier for the item. For example, you can include the ribbon group name in the ID.
   - In the Icon field, enter the path to the relevant icon. Depending on the button you create, adjust the icon size accordingly.
4. Open Sitecore Rocks and add the relevant control rendering, for example SmallButton, to the layout of the button item you created.



5. Enter a unique ID for the rendering.
6. For other SPEAK controls, you can point to another item in the Data Source field and specify the configuration in this other item.

   Important

   For this rendering, do not specify a placeholder.

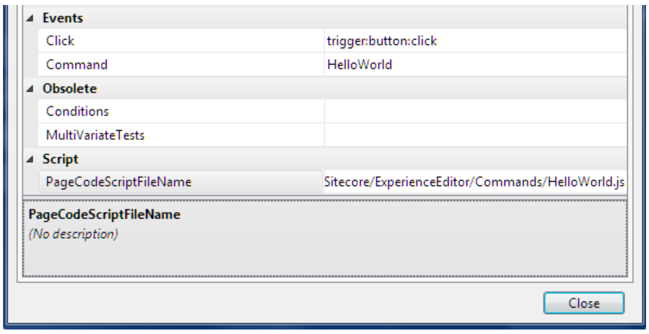
Assign a command to the button

To create and assign a command for the new button:

1. In the folder structure of your Sitecore installation, navigate to *\Website\sitecore\shell\client\Sitecore\ExperienceEditor\Commands* and create a JavaScript file, for example, `helloworld.js`.
2. Open the file and insert the relevant code. For example:

```
define(["sitecore"], function (Sitecore) {
  Sitecore.Commands.HelloWorld =
  {
    canExecute: function (context) {
      // Determines whether command is disabled or enabled.
      return true;
    },
    execute: function (context) {
      alert("Hello world!");
    }
  };
});
```

3. In Sitecore Rocks, right-click the new button item, click Task, and then Design Layout.
4. Double-click the SmallButton rendering and in the Edit Rendering Properties dialog box, change the following values:
   - For the Click property, set the value to “trigger:button:click”.
   - For the Command property, enter the command object name defined in Sitecore.Commands namespace, for example *HelloWorld*.
   - In PageCodeScriptFileName field, enter the path to your JavaScript file that must execute.



5. Click Close and open the Experience Editor to verify that your button appears in the ribbon.

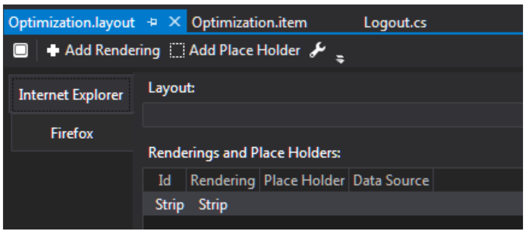
Hide an existing ribbon control

In the Experience Editor ribbon, you can hide the existing buttons or tabs:

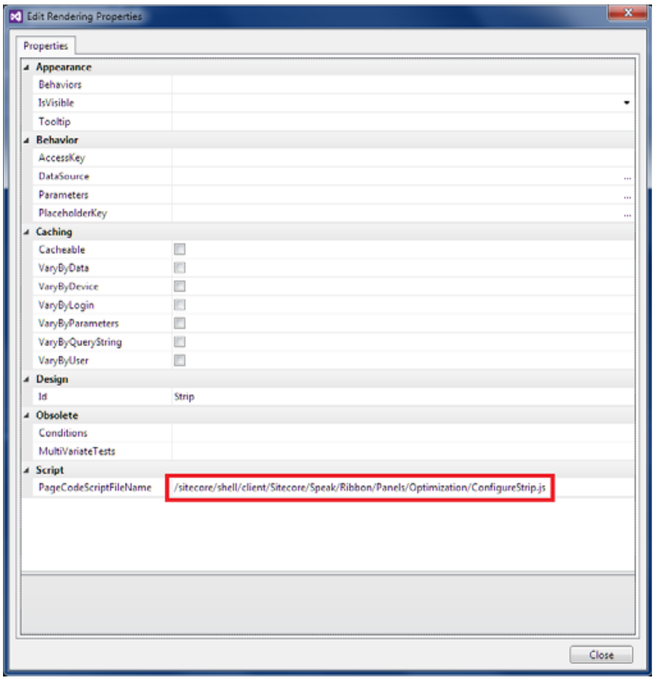
- To hide existing buttons, use the security settings.
- To hide an existing tab, you must first create a JavaScript file that hides the tab when it is executed and then assign it to the tab that you want to hide.

To hide an existing tab:

1. In the folder structure of your Sitecore installation, navigate to *\Website\sitecore\shell\client\Sitecore\ExperienceEditor\Commands* and create a JavaScript file that, when executed, hides the tab.
2. In Sitecore Rocks, navigate to the tab that you want to hide and click Task and then click Design Layout.
3. In the design layout for the tab that you want to hide, in the Renderings and Place Holders section, double-click the Strip rendering.



4. In the Edit Rendering Properties dialog box, in the PageCodeScriptFileName field, enter the path to the custom JavaScript file that hides the tab, when it is executed.



Send feedback about the documentation to docsite@sitecore.net.

Navigate to an item in the Experience Editor

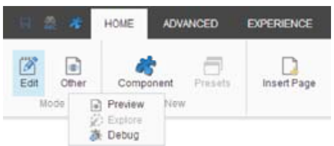
In the Experience Editor, you can navigate to the page or item that you want to work on using one of the following methods:

- Preview mode – use this method if you want to browse the website to get to the page.
- Navigation bar – use this method if you want to navigate to the page in a menu structure.
- Search – use this method if you want to search for the specific item. This is very helpful if you need to work on an item or page on another website.

Preview mode

If you enable the preview mode, you can browse to a specific page like on a regular website.

- To enable the preview mode, in the ribbon, in the Mode group, click Other, and in the drop-down menu, click Preview.

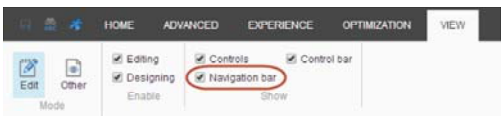


You can now click the items on your website to browse to the relevant page.

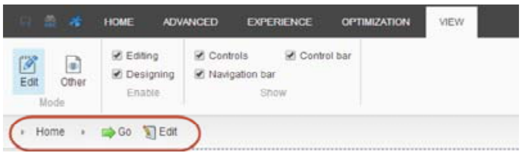
Navigation bar

The Navigation bar allows you to navigate to a specific page through a menu structure.

- To enable the Navigation bar, in the ribbon, on the View tab, select Navigation bar.



The Navigation bar appears below the ribbon.

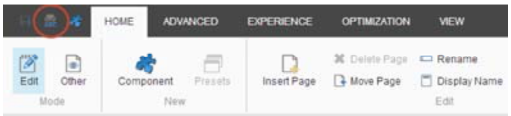


- To navigate to a page, click the item or the arrow in the Navigation bar to navigate to a page in the menu. If you want to edit the item's field values in the Content Editor, click Edit .

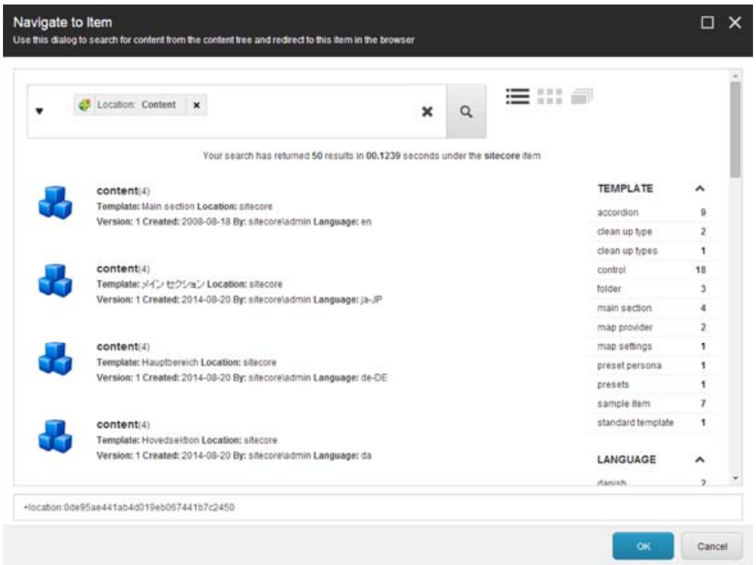
Search

In the Experience Editor, you can use search to navigate to a specific item and from the search results direct the Experience Editor to the specific page where the item is located. In addition, if you work on multiple websites, you can use search to quickly move from one website to another.

- To use search to redirect the Experience Editor to a specific item, click Search in the Experience Editor ribbon.



In the Navigate to Item dialog box, you can search for an item across all your websites and direct the Experience Editor to that specific item.

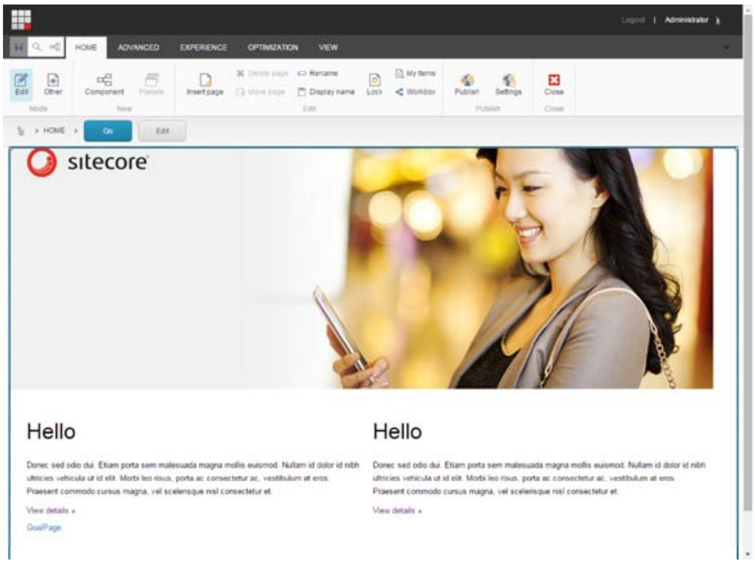


Send feedback about the documentation to docsite@sitecore.net.

The Experience Editor

The Experience Editor is a WYSIWYG editor that allows you to easily make changes to items directly on the page. You can edit all the items that are visible on the page — text, graphics, logos, links, and so on.

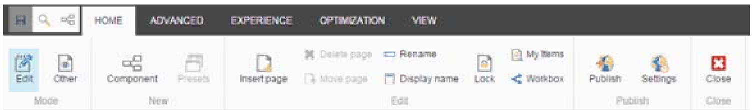
In the Experience Editor, you can edit not only the fields of the current item, but also the fields of any items rendered on the page. For instance, if the menu on the page consists of titles of the product items, you can edit the titles without navigating to the product items themselves.



The ribbon

In the Experience Editor, the ribbon with all the available functionality is displayed at the top of the webpage. The functionality in the ribbon vary depending on your security roles and the customizations of the website that you are accessing.

If you have full access to all functionality in the Experience Editor, you have access to all five tabs in the ribbon, each with its own collection of commands.

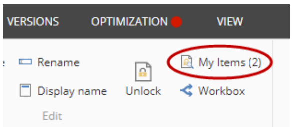


As a minimum, you can expect to be able to insert a page and delete a page on your website. Regardless of the ribbon that you use, there is always a Save button in the top-left corner.

When you edit an item, you can click the Toggle Ribbon button to hide the ribbon temporarily and give yourself some more space to work with. Click the Toggle Ribbon button again to display the ribbon.

Improve the ribbon load time

You can improve the time it takes to load the Experience Editor ribbon if you turn off the request that displays the number of items you have locked.



To improve the ribbon load time:

- In the `Website\App_Config\Sitecore\Experience Editor\` folder, open the `Sitecore.ExperienceEditor.config` file. In the `WebEdit.ShowNumberOfLockedItemsOnButton` setting, change the value to *false*.

Send feedback about the documentation to docsite@sitecore.net.

Add a search result view

Sitecore can show search results in the following different views: ID, Image, Lock, Media, Preview, Table, and Tag view. You can add new views if these views do not fit your needs.

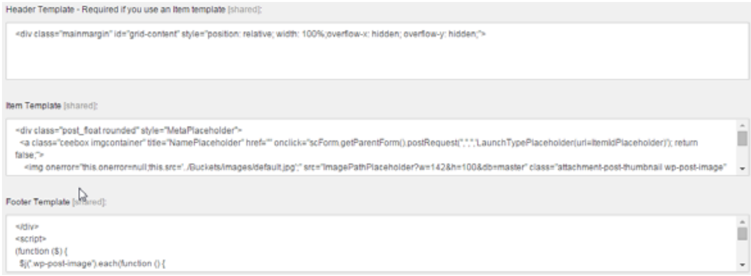
This topic outlines how to:

- Add a new search result view
- Show a dynamic field in a search result view

Add a new search result view

To add a new view:

1. Navigate to `/sitecore/system/Settings/Buckets/Views` and in the *Views* folder, create a new view item.
2. In the Header Template, Item Template, and Footer Template fields, specify placeholders and HTML tags to format the search result:



You can use the following built-in placeholder names to display item values in the search result:

Placeholder	Description
MetaPlaceholder	The CSS style that you want to use when the results are displayed.
LaunchTypePlaceholder	Specifies the result is launched in a new tab or in a new Content Editor window.
ItemIdPlaceholder	The item ID.
ImagePathPlaceholder	The path to the image of the item.
NamePlaceholder	The name of the item.
TemplatePlaceholder	The name of the template that the item is based on.
BucketPlaceholder	The bucket that this result comes from.
ContentPlaceholder	The content of the result.
VersionPlaceholder	The version of the content item.
CreatedPlaceholder	The date that the content item was created.
CreatedByPlaceholder	The person who created this item.

Show a dynamic field in a search result view

You show dynamic fields in search result views by specifying the field in the `buckets.dynamicFields` pipeline. For example, if you want to display Facebook likes for a specific item you can use this pipeline to display this information in the search results.

The `buckets.dynamicFields` pipeline aggregates a dictionary of keys and values. To obtain the information you want, you must refer to the relevant key and value in the dictionary. In the Facebook example, the key is `Flikes` and the value is the number of likes specified by the Facebook API.

To display a dynamic field in a specific search view:

1. In the content tree, navigate to `/sitecore/system/Settings/Buckets` and in the *Views* folder, select the search view.
2. On the Content tab, in the View Details section, click the Item Template field.

   This field contains the HTML output for the search view.

3. To insert the placeholder, enter the relevant key followed by `DynamicPlaceholder`.

   For example, if the key for Facebook likes is `Flikes` in the dictionary, the placeholder is: `FlikesDynamicPlaceholder`.

When Sitecore displays the search results, it looks at all the dynamic placeholders and replaces them with the value of each specific key.

Send feedback about the documentation to docsite@sitecore.net.

Add new search filters and alias filters

The search filters describes how to use search filters to build queries. You can pick from a number of existing filters. Sitecore provides a number of search filters by default, but you have the ability to add more custom filters.

To add a new search filter:

1. In the content tree, navigate to `/sitecore/system/Settings/Buckets/Search Types`. Sitecore recommends that you add new search filters in the *User Defined* folder.

2. Select the *User Defined* folder, and on the Home tab, in the Insert group click Field Search Type. Give the new search type an appropriate name. This is the name that users enter in the search box to apply the filter.

   For example: if you name the search type *Date*, the user must type in *Date:* in the search box to apply the filter. The name is case sensitive.

3. In the new search type item, in the Search Type section, in the Control Type field select the type of control that is most appropriate for your search.
4. In the Display Text field, enter an appropriate text and ensure that the wording is consistent with all the other search filters. This text is displayed in the drop-down menu when users browse the search filters.
5. If you want to apply a custom syntax to the Control Type field to create specific output., enter this in the Web Method field.

   For example, if the Control Type field is a calendar, the Web Method field can make a request to a web service to tell Sitecore to display a calendar control that only allows you to select a date from the last 2 calendar years.

Alias filters

Some field names are long and descriptive, and therefore not very easy to type into a search box.

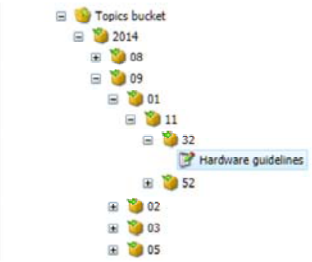
You can create alias filters. These work as aliases for search fields with long names to make searching easier. For example, you could assign an alias to a field called Product Price, giving it the more searchable alias price. You can also add a slider control so users can slide between prices. The control type parameters for this slider are:

```
min:0&max:20000&value:40&range:true&start:0&end:2000
```

Send feedback about the documentation to docsite@sitecore.net.

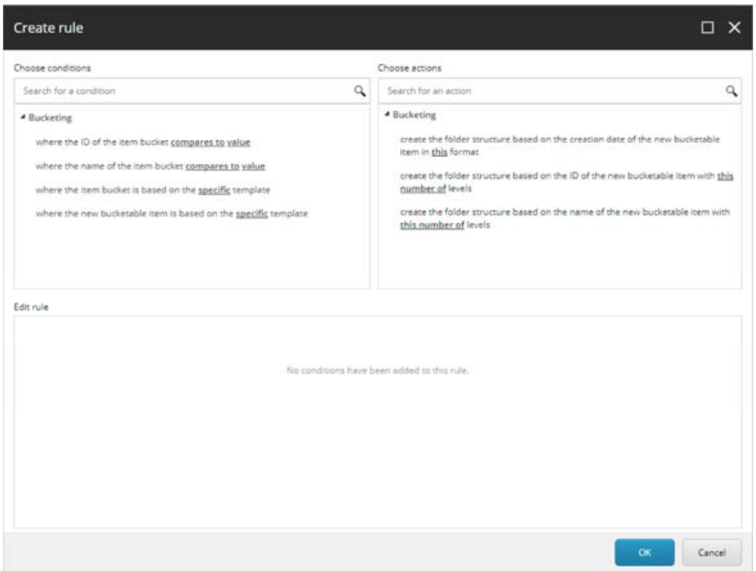
Custom bucket structures

The *BucketConfiguration.BucketFolderPath* configuration setting defines the structure Sitecore uses for showing items buckets as folder trees. The default setting creates a structure based on the date and time – year, month, day, hour, and minute – that the item was created:



To create a custom bucket structure:

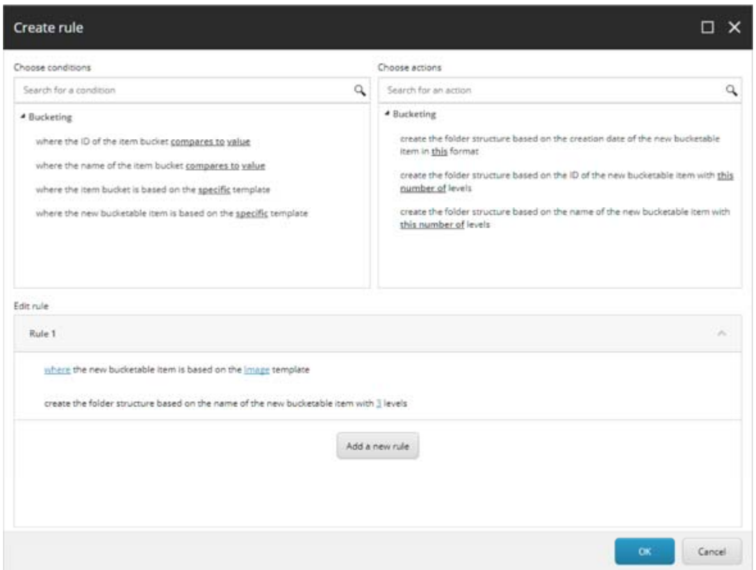
1. Navigate to the *sitecore/System/Settings/BucketItem Buckets Settings* item.
2. Specify the rules in the Rules for Resolving the Bucket Folder Path field:



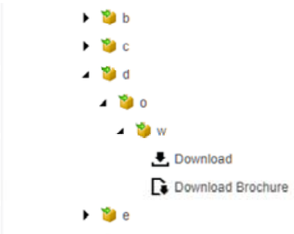
3. Create rules by selecting a condition (such as “where the bucket item template is…”) and an action.

   The action specifies the bucket folder structure. Only bucket items that match the condition will use the action. You can create any number of rules. Sitecore uses the format defined by the *BucketFolderPath* configuration setting for items that do not match any condition.

   For example, this rule:



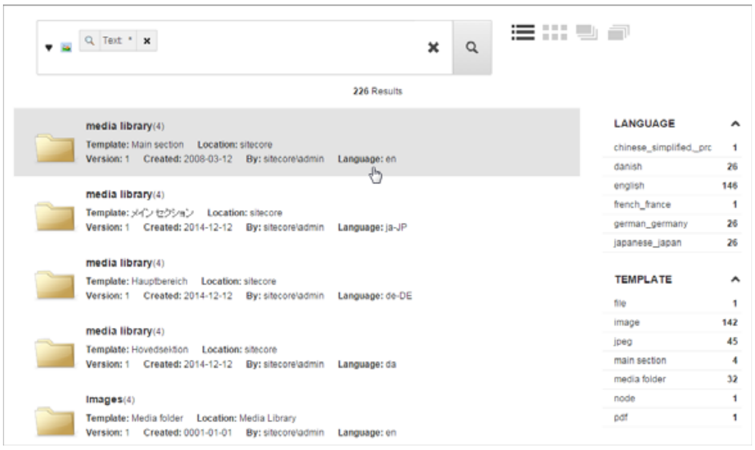
creates this bucket structure:



Send feedback about the documentation to docsite@sitecore.net.

Facets

Facets group and classify content items, for example, by the language that they are in, the date on which they were created, or the template they are based on. Users use facets to filter and limit search results after running searches. Sitecore shows the facets of a search result to the right of the search result itself:



In this example, one of the facets is called Language, and this facet has the following filters:

- chinese_simplified_prg
- danish
- english
- french_france
- german_germany
- japanese_japan

When users click, for example, the `danish` filter, only items where the Language facet has the value `danish` will be shown.

For any given facet (such as Language), Sitecore creates the filters (such as `danish`) dynamically. The filters are based on data, and they change if and when data changes.

Sitecore only shows a facet and its filters when it is relevant to the actual search result. This behavior is dynamic.

Search facets in Sitecore

Sitecore provides these facets and uses them, for example, when users search in the Content Editor:

Facet	Description
Author	Groups the results according to the authors who created the items.
Author Template	Groups the results according to a combination of author and template.
Bucket	Groups the results according to the buckets that they are stored in.
Creation Date & Author	Groups the results according to the date that items were created and who authored them.
Date Range	Groups the results in date ranges: Today, Yesterday, 2-4 days ago, and so forth.
File Size	Groups the results according to the size of the file: small, medium, or large.
File Type	Groups the results according to file type.
Image Dimensions	Groups the results according to the dimensions of the images they contain.
In Workflow	Groups the results according to the workflow they are in.
Language	Groups the results according to language.
Language Template	Groups the results according to the languages that the templates are in.
Location	Searches all the bucket locations and groups the results according to which buckets the results are stored in.
Tags	Groups the results according to their tags.
Template	Groups the results according to templates.
Template Author	Groups the results according to a combination of template and author.
Updates Date	Groups the results according to the date that items were last updated.

Creating and configuring facets

Facets are stored at: `/sitecore/system/Settings/Buckets/Facets`. You can create a facet by right-clicking on a Facets item and selecting Insert, Facet in the context menu and then specifying a name for your new facet.

You can create folders for your facets. This can make it easier to get an overview of the facets and help separate the facets you create from the facets that Sitecore delivers.

You can configure facets you create in two different ways:

- You can specify item field names. When you specify a field name in the Field Name field of the facet item, Sitecore creates a facet based on this field. You can enter multiple field names as a comma-separated list and when you do, Sitecore creates hierarchical filters (for example, `belts/black`, `belts/white`, and so on.) You specify the field name in lower case, and use the name of the field as it is in the index.

  Fields that can have many different values across items are not good candidates for facets, and you should avoid using such fields. For example, a field that has a unique value in each item would cause Sitecore to create a filter for each item.

- You can specify a reference to a class that implements the ISimpleFacet interface in the Facet Filter field. You have to create this class yourself, and it has to return a string. You can implement logic in this class that determines the filters that the facet should search for.

  The intention is that this class reduces the number of filters. This is how some of the Sitecore search facets are implemented, for example, Date Range or File Size.

Global and local facets

Facets can be global or local. Global facets are used for all searches in all parts of the content tree. The search facets delivered by Sitecore are global facets. To specify that a facet is a global facet:

- Select the Global Facet field in the Facet section of the facet.

Local facets are used for a specific item bucket or for an item in the content tree (and the items that are descendants of this item). To specify facets for an item bucket or an item:

- Select facets in the Facets field of the Indexing section of the item bucket or item.

Using facets in SPEAK applications

You can create and use facets in SPEAK applications with the FilterControl component.

Send feedback about the documentation to docsite@sitecore.net.

Joining data

A join – or a join statement – combines records from two or more tables in the database. This topic describes when to use joins and what you must consider when you use them.

The LINQ layer supports the following four joins: `Join`, `Self Join`, `Group Join`, and `Select Many`. These four methods use subqueries that are evaluated as users page through data. Lucene.Net does not support these join methods natively, and Solr only supports certain aspects.

Important

You must try to avoid using these methods when you can. They are all bad for performance, both for processing time, amount of I/O, and for memory usage, and this is true for both Lucene and Solr.

For example, you use a `Join` query with LINQ to join *Contacts* with the *Engagement States* they have been in. When users page through the data, the join or subquery is *only* executed for the first page. That is: given an ID, a subquery is run to look up another document via that ID.

When you execute a `Join` query in LINQ, it effectively returns a union set of one or many documents. You must consider performance when you write LINQ queries, in the same way as when you write SQL queries with joins. You must also store and index your data properly. Sitecore does not support many-to-one or many-to-many queries in LINQ (for example, a *Contact* that stores multiple references to *Engagement States* and then joins those on the *Engagement States* documents).

You can attach the “foreign-key” to the *Engagement State,* so that you can join using something like the *ContactId* instead:

```
public class Contact {
public string Name { get; set; }
public Guid ContactId { get; set; }
}
public class EngagementState {
public string Name { get; set; }
public Guid ContactId { get; set; }
public Guid Id { get; set; }
}
```

Given the classes above, and even though it results in duplication of data, this lets you join *Contacts* to *Engagement States* via the *ContactId*. The LINQ layer supports this, for example:

```
var repo = this.CreateVisitors();
var repoPlans = this.GetStates();
var result = from t in repo
join x in repoPlans on t.ContactId equals x.ContactId
where x.Id == new Guid("E1B604F1-EE0E-408E-A344-869CC45D25D9")
select t;
```

Testing in the LINQScratchPad

You can test joins on large amounts of data in the LinqScratchPad.aspx, like this:

```
using (var context = ContentSearchManager.GetIndex("sitecore_master_index").CreateSearchContext())
{
    using (var context2 = ContentSearchManager.GetIndex("sitecore_web_index").CreateSearchContext())
        {
            return context.GetQueryable<SearchResultItem>().Join(context2.GetQueryable<SearchResultItem>()
            .Where(i => i.Name.StartsWith("S")), i => i.ItemId, o => o.ItemId, (o, i) => o).Take(10).ToList();
        }
}
```

This opens two separate search contexts and then runs a join on them, based on the *ItemId*. In this example, this query checks which items where the name field starts with an S are in both the `web` and the `master` index, and then returns the results of the `web` index (*outer*).

Joins in Solr

When you use the Solr provider, `Self Join` is the only join that runs a real Solr join. The other methods (such as `Join` and `Group Join`) use the enumeration technique that the Lucene.net provider uses: at enumeration time, subqueries are executed to get the other documents.

When to use joins

You need to use a join in the LINQ layer when you have a document that contains a reference to another document. The reference is typically an ID reference.

Sitecore does not flatten the objects it crawls. The crawler implementation has to tell Sitecore how to store data. The `Join`, `Self Join`, or `Group Join` methods run a subquery for the join to get another document and evaluate it based on the ID/Key.

You must make sure that you store your data properly. You can prepare your data so that it is ready for joining, or you can prepare it for a completely join-free retrieval. Both approaches have advantages as well as disadvantages.

Follow these rules to ensure that you solution scales well:

- Limit the amount of joins.
- Consider flattening data instead of using a join. This results in multivalued fields and potentially many columns.
- If you use a join, keep your paging of the data small – 10 or 20 items at a time. The reason is that if you have a query that takes 100 milliseconds to run, it could take 1.1 seconds to return the subqueries and the initial query.
- Only store or index what is 100% necessary (unless this has performance implications).

The differences between Join, Self Join, and Group Join methods

For example, if you have the following two tables of data:

ID	Value
1	A
2	B
3	C

ID	ChildValue
1	a1
1	a2
1	a3
2	b1
2	b2

When you use the `Join` method on the two lists on the `Id` field, the result is:

Value	ChildValue
A	a1
A	a2
A	a3
B	b1
B	b2

If you use the `Group Join` method on the two lists on the `Id` field, the result is:

Value	ChildValue
A	[a1, a2, a3]
B	[b1, b2]
C	[]

If you use the `Self Join` method on the two lists on the `Id` field, the result is:

Value	ChildValue
A	a1
A	a2
A	a3
B	b1
B	b2

If you use the `Self Join` method, you do not need to provide a *resultSelector* parameter. It is inferred that it returns the *outer* result. This is essentially the difference between an inner and outer join.

Note

When you run join queries, your *inner* or *outer* joins should have a filter that limits the results. Otherwise, the join queries will try to join on every single document in your index.

Send feedback about the documentation to docsite@sitecore.net.

Language specific search

The Search API has two elements you can use to implement content search that only searches in one language: *CultureExecutionContext* and *InContext*.

This example shows how you can use them:

```
var index = ContentSearchManager.GetIndex("sitecore_master_index");
using (var context = index.CreateSearchContext())
{
    var cultureZh = new CultureExecutionContext(CultureInfo.GetCultureInfo("zh-TW"));
    var result = context.GetQueryable<TestSearchResultItem>(cultureZh)
.Where(it => it.Title == "Home").ToArray();
    var result2 = context.GetQueryable<TestSearchResultItem>()
                 .InContext(cultureZh)
                 .Where(it => it.Title == "Home").ToArray();
}
```

Where:

```
<field fieldName="title" returnType="text" />
public class TestSearchResultItem : SearchResultItem
{
    public string Title { get; set; }
}
```

Send feedback about the documentation to docsite@sitecore.net.

LINQ to Sitecore

LINQ to Sitecore provides access to search the indexes, using standard LINQ queries in the same way that other LINQ providers, such as LINQ to SQL and LINQ to Objects, work. It uses the standard `IQueryable<T>` interface and supports most of the available operations. For a general introduction to LINQ, see http://msdn.microsoft.com/en-us/library/vstudio/bb397926.aspx.

Sitecore supports two search providers: Lucene and Solr. The LINQ layer is an abstract layer that converts common queries to something that these search providers understand.

For example, the LINQ layer resolves a query like this:

```
var query = context.GetQueryable<Product>.Where(item => item.Name == “Sample Item”)
```

to something that Solr or Lucene understands. If you implement a new search provider, this provider can also understand the query.

The LINQ layer is used internally by Sitecore but developers can also use it. You can use this layer in sublayouts.

To start a search, you set up a search context:

```
using (var context = ContentSearchManager.GetIndex(item).CreateSearchContext())
```

```
{
    IQueryable<SearchResultItem> searchQuery = context.GetQueryable<SearchResultItem>().Where(item => item.Name == "Sitecore")
}
```

The LINQ layer converts the query to something your provider understands. For example, for Lucene it is converted to:

```
_name:sitecore
```

This returns the results of a query on your search index and returns it as a `SearchResultItem` type. You can also use the indexer to run queries:

```
using (var context = ContentSearchManager.GetIndex(item).CreateSearchContext())
{
    IQueryable<SearchResultItem> searchQuery = context.GetQueryable<SearchResultItem>().Where(item => item["_name"] == "Sitecore")
}
```

This topic describes:

- Supported IQueryable methods
- LINQ to Sitecore syntax
- IQueryable extensions
- Custom search type/object mapping

Supported IQueryable methods

The LINQ layer does not implement all `IQueryable` methods. The following methods have been implemented:

- Sort by standard string, integer, float, or any type that implements `IComparable`
- All
- Any
- Between (with an extra overload for including or excluding the start and end ranges)
- Boost (makes this part of the query more important than the other parts)
- Cast (you can use this instead of Select)
- Contains
- Count
- ElementAt
- EndsWith
- Equal
- Facets (an extension that fetches the facets of predicates)
- First
- FirstOrDefault
- Join
- Last
- LastOrDefault
- Min
- Max
- Match (an extension for running regular expression queries)
- OrderBy
- OrderByDescending
- Select
- Single
- SingleOrDefault
- Skip
- Take
- ToList()
- ToLookUp()
- ToDictionary()
- Page (an extension that does Skip and Take automatically for you)
- StartsWith

Not supported IQueryable methods

These methods are not supported:

- GroupBy
- GroupByJoin
- Intersect
- Sum
- Average
- Concat
- TakeWhile
- SkipWhile
- Reverse
- Union

If these methods are called, a `NotSupportedException` or `InvalidOperationException` exception is thrown at runtime.

LINQ to Sitecore syntax

This tables shows how the LINQ layer and Lucene correspond:

	Lucene syntax	LINQ to Sitecore
Fields	`text:"go"`	`c.Text == "go"` or

		`c.Text.Equals("go")`
WildCard	`text:*amber*`	`c.Text.Contains ("amber")`
Prefix	`text: *amber`	`c.ContactName.StartsWith("amber")`
Fuzzy	`text:roam~` `text:roam~0.8`	`c.Text.Like("roam")` `c.Text.Like("roam", 0.8)`
Proximity	`text:"jakarta apache"~10`	`c.Text.Like("jakarta apache", 10)`
Inclusive Range	`__smallupdateddate:[20140101 TO 20150101]`	`.Where(c => c.Updated.Between(new DateTime(2014, 1, 1), new DateTime(2015, 1, 1), Inclusion.Both)`
Exclusive Range	`title:{Aida TO Carmen}`	`c.Title.Between("Aida", "Carmen", Inclusion.None)`
Boosting	`name:jakarta^4 apache`	`.Where(c => c.Name.Equals("jakarta").Boost(4) || c.Name.Equals("apache"))`
Boolean Or	`name:"jakarta apache" OR jakarta`	`where c.Name.Equals("jakarta apache") || c.Equals("jakarta")`
Boolean And	`"jakarta apache" AND "Apache Lucene"`	`where c.Equals("jakarta apache") && c.Equals("Apache Lucene")`
Boolean Not	`"jakarta apache" NOT "Apache Lucene"`	`where c.Equals("jakarta apache") && !c.Equals("Apache Lucene")`
Grouping	`(jakarta OR apache) AND website`	`where (c.Title == "jakarta" || c.Title == "apache") && (c.Title == "website")`

IQueryable extensions

The LINQ layer provides extra methods that extend the standard `IQueryable` interface. You must declare the `Sitecore.ContentSearch.Linq` namespace to use them.

Filtering

Filtering is similar to using `Where` to restrict the result list. When you use `Filter`, the scoring/ranking of the search hits is not influenced by the filters, and filters can be cached to optimize search performance.

For example:

```
results = queryable.Filter(d => d.Id > 4 && d.Id < 8);
```

Note

To avoid influencing the ranking of the search results, use `Filter` when applying restrictions to search queries in the `GetGlobalFilters` pipeline.

Facets

Simple faceting

```
var results = queryable.FacetOn(d => d.Name);
var facets = results.GetFacets();
foreach (var category in facets.Categories) {
    Console.WriteLine(category.Name);
    foreach (var facetValue in category.Values) {
        Console.WriteLine("{0}: {1}", facetValue.Name, facetValue.Aggregate);
    }
}
```

Pivot faceting

```
var results = queryable.FacetPivotOn(p => p.FacetOn(d => d.Name).FacetOn(d => d.Year));
var facets = results.GetFacets();
foreach (var category in facets.Categories) {
    Console.WriteLine(category.Name);
    foreach (var facetValue in category.Values) {
        Console.WriteLine("{0}: {1}", facetValue.Name, facetValue.Aggregate);
```

```
    }
}
```

Boosting

```
queryable.Where(it => (it.TemplateName == "Sample Item").Boost(50) || it.TemplateName=="Folder");
queryable.Where(it => (it.Paths.Contains(new ID("{0DE95AE4-41AB-4D01-9EB0-67441B7C2450}")).Boost(3) || it.TemplateName=="Folder") );
```

Other

Between

```
results = queryable.Where(item => item.Price.Between(50.0f, 400.0f, Inclusion.Both));
results = queryable.Where(item => item.Price.Between(2.0f, 12.0f, Inclusion.Both) || item.Price.Between(80.0f, 400.0f, Inclusion.Both)
results = queryable.Where(d => d.Date.Between(new DateTime(2004, 12, 31), DateTime.Now, Inclusion.Both));
results = queryable.Where(d => d.Id.Between(1, 4, Inclusion.Both));
results = queryable.Where(d => d.Id.Between(1, 4, Inclusion.Lower));
results = queryable.Where(d => d.Id.Between(1, 4, Inclusion.Upper));
results = queryable.Where(d => d.Id.Between(1, 4, Inclusion.None));
```

string.Contains

```
results = queryable.Where(d => !d.Template.Contains("Hello:));
```

string.CompareTo

```
results = queryable.Where(d => !d.Name.CompareTo("Hello") == 1);
```

Equal

```
results = queryable.Where(d => d.Id.Equal(4));
```

Matches

```
results = queryable.Where(i => i.Template.Matches("^.*$"));
```

MatchWildcard

```
results = queryable.Where(i => i.Template.Where(i => i.Template.MatchWildcard("H?li*m")));
```

Like

```
results = queryable.Where(i => i.Template.Like("Citecoar"));
```

string.StartsWith

```
results = queryable.Where(d => !d.Name.StartsWith("Hello"));
```

string.EndsWith

```
results = queryable.Where(d => !d.Name.EndsWith("Hello"));
```

GetResults

```
results = queryable.GetResults().Hits.Where(i => i.Document.Name.Contains("o")).Where(hit => hit.Score > 0.6);
```

GetFacets

```
results = queryable.Where(d => d.Id > 0).FacetOn(d => d.Template, 0).GetFacets();
```

Custom search type/object mapping

Because the LINQ layer uses the generic `IQueryable<T>` interface to expose the search indexes, you can use custom classes or POCO classes to describe the information in the indexes.

To implement custom search types, such a class must:

- Have an empty constructor.
- Expose public properties with getters and setters and/or a public indexer to hold the search result data.

The LINQ provider automatically maps document fields in the index to properties on the custom search type by the names of the properties. Properties or fields from the index that have not been matched together by name are skipped during the object creation/mapping.

It is also possible to map properties that do not match to fields in the index by decorating the properties with the `IndexField` attribute. You can use this, for example, to expose special Sitecore fields such as `_name` as a property called `Name`. A different use case is field names with spaces, because they cannot be mapped directly to a property by name.

Furthermore, you can implement an indexer that is populated with the field name as key and the value for each field in the index document. There is also an `ObjectIndexerKey` that you can use to wrap indexers as different types. This is useful if you only have the string version of a property name but need to use it as an indexer for a property type, where you most often need an int.

The process of supplying a custom type and let Sitecore map from index fields to the properties of the item is also known as "hydration".

Depending on the search provider being used, the indexed and stored data in the index might not be the native types for the value. For Lucene, everything is stored and indexed as strings.

Supported types

The following types are supported for automatic type conversion when mapping index document fields to properties:

- .NET built-in integral types
- .NET built-in floating-point types
- Boolean
- String
- DateTime
- Guid
- Sitecore ID
- Sitecore ShortID
- Sitecore ItemUri
- IEnumerable<T>
- DateTimeOffset
- Language
- Version
- Database
- CultureInfo
- TimeSpan

Custom search type example

The following is a short example of how you implement a custom search type:

```
public class MySearchResultItem
{
// Fields
private readonly Dictionary<string, stringfields = new Dictionary<string, string>();
// Properties
// Will match the _name field in the index
[IndexField("_name")]
public string Name { get; set; }
// Will match the myid field in the index
public Guid MyId { get; set; }
public int MyNumber { get; set; }
public float MyFloatingPointNumber { get; set; }
public double MyOtherFloatingPointNumber { get; set;}
public DateTime MyDate { get; set; }
public ShortID MyShortID { get; set; }
public ID SitecoreID { get; set; }
// Will be set with key and value for each field in the index document
public string this[string key]
{
get
{
return this.fields[key.ToLowerInvariant()];
}
set
{
this.fields[key.ToLowerInvariant()] = value;
}
}
}
```

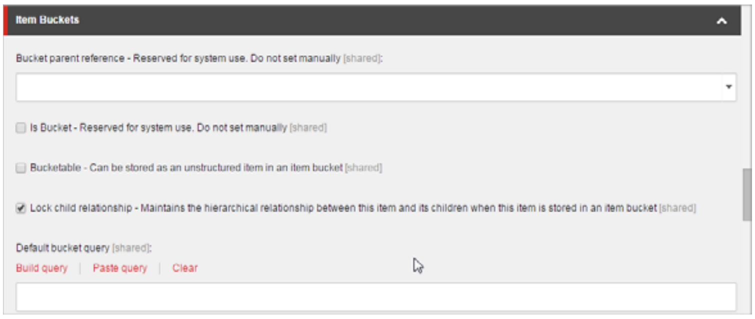
Send feedback about the documentation to docsite@sitecore.net.

Lock a parent-child relationship

In some cases, you want to lock the relationship between a parent item and its child items even though both are stored in an item bucket. You might need to ensure that the child items are always stored below the parent item, for example, by locking the parent to child relationship between news articles and comments.

To lock a parent to child relationship:

1. In the Template Manager, navigate to the template for the parent item. For example, the news article template.
2. Expand the template in the content tree, and select the *_Standard Values* item.
3. In the right pane, scroll down to the Item Buckets section.



4. Select the Lock child relationship check box.

   Note

   If you create a content item that is a child of a content item based on this template, it is not automatically structured in the item bucket. Instead, it retains its relationship with the parent item. For example, comments will always be children of the news article that they refer to.

Send feedback about the documentation to docsite@sitecore.net.

Queries for null or empty strings

The nature of analytics data means that you sometimes want to run queries for null or empty string values.

Sitecore supports this in the LINQ layer and in the indexing layer by storing physical values of `EMPTYVALUE` and `NULLVALUE`. Sitecore translates this in the LINQ layer, so all you need to do is write LINQ queries as usual.

You must specify which fields you want to have an empty string and null support for in the FieldMap. You should not store empty strings and nulls in your index because they waste space.

However, there are some cases where you need to do this. For example, when you want users to be able to search for all customers that have not specified a gender and to contact these customers to determine this. To make this possible, you must set up the FieldMap to store empty strings in the `Gender` field.

For Lucene, you do this in the `App_Config\Include\Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config` file in the `<fieldNames hint="raw:AddFieldByFieldName">` section.

For example:

```
<field fieldName="gender" storageType="NO"  indexType="TOKENIZED"    vectorType="NO" boost="1f" type="System.String" nullValue="NULLVA
```

For Solr, you do this in the `App_Config\Include\Sitecore.ContentSearch.Solr.Indexes.config` file in the `<fieldNames hint="raw:AddFieldByFieldName">` section.

For example:

```
<fieldType fieldName="title" returnType="text" nullValue="NULLVALUE" emptyString="EMPTYVALUE"/>
```

```
<fieldType fieldName="title_t" returnType="text" nullValue="NULLVALUE" emptyString="EMPTYVALUE"/>
```

When you run the following query, it maps `""` to `EMPTYVALUE` and null to `NULLVALUE`:

```
return context.GetQueryable<Contact>().Where(i => i.Gender == “”).Take(10).ToList();
```

Send feedback about the documentation to docsite@sitecore.net.

Supporting autocomplete with n-gram

Sitecore provides an n-gram analyzer for Lucene.net (`Sitecore.ContentSearch.LuceneProvider.Analyzers`). If you use Solr, you can set this up in the `Solr Schema.xml` file.

You use the n-gram analyzer to create autocomplete functionality for search input. The analyzer breaks tokens up into unigrams, bigrams, trigrams, and so on. When a user types a word, the n-gram analyzer looks the word up in different positions, using the tokens that it generated.

You add support for autocomplete by adding a new field to the index and mapping this field to use the n-gram analyzer instead of the default. When you run the LINQ query to query that field, use the following code:

```
using (IProviderSearchContext context = Index.CreateSearchContext())
            {
                result = context.GetQueryable<SearchResultItem>().
                    .Where(i => i.Name.StartsWith(“some”))
                    .Take(20)
                    .ToList();
            }
```

Sitecore provides an implementation that uses trigrams and a set of English stop words. If you have other requirements, you can build a new analyzer and change these settings.

Send feedback about the documentation to docsite@sitecore.net.

The IndexCrawler

The `IndexCrawler` is a crawler that takes an existing index and executes a group of functions over it.

The `IndexCrawler` inherits from the `FlatDataCrawler<T>`. This means that you can pass a list of `IIndexables` and it iterates over them, then executes a function on each `IIndexable` and commits it back to the index.

This an example of what you can use it for: Go through every document in the index and add “!!” to the name of every document. The code would look like this:

```
var crawler = new IndexCrawler(this.sourceIndexName);

crawler.CrawlFunctions += this.CallMe;

this.DestinationIndex.AddCrawler(crawler);

this.DestinationIndex.Rebuild();

public IEnumerable<IIndexable> CallMe(IProviderSearchContext context)

{

    var list = new List<IIndexable>();

    foreach (var item in context.GetQueryable<InsertDocument>())

    {

        item.Name = item.Name + "!!";

        list.Add(new ObjectIndexable(item, null));

    }

    return list.AsEnumerable();

}
```

Send feedback about the documentation to docsite@sitecore.net.

Using ObservableAggregator<T> to filter items

You can configure instances of `ObservableAggregator<T>` processors that filter out items that you do want to be not be sent to observers. All observables use the global `aggregation.filter.inbound` pipeline. You can add additional observable-specific pipelines.

Configuration

You implement the global aggregation filter in XML. The following code adds two filters. The first one filters by contact ID and the second one by the path visited:

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">  <sitecore>

    <pipelines>

      <!-- global aggregation inbound filters-->

      <aggregation.filter.inbound>

        <processor type="Sitecore.ContentSearch.Analytics.Pipelines.AggregationFilters.InboundContactIdFilter, Sitecore.ContentSearch.A

          <filters hint="list:AddFilter">

            <filter>ca82109f-30cd-4414-956e-caea41c7510c</filter>

          </filters>

        </processor>

        <processor type="Sitecore.ContentSearch.Analytics.Pipelines.AggregationFilters.InboundVisitPathFilter, Sitecore.ContentSearch.A

          <filters hint="list:AddFilter">

            <filter>/default.aspx</filter>

          </filters>

        </processor>

      </aggregation.filter.inbound>

    </pipelines>

  </sitecore>

</configuration>

You can configure each processor in the sample above with additional <filter> values.
```

Observable specific filter

You can configure an observable-specific pipeline in addition to the global filter pipeline. If you configure an observable called `DefaultObservable`, you simply configure a pipeline called `defaultobservable.filter.inbound`. For example:

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">

  <sitecore>
```

```
    <pipelines>
      <!-- define the aggregator -->
      <aggregation>
        <processor patch:before="*[1]"
                   type="Sitecore.ContentSearch.Analytics.AnalyticsObservableAggregator, Sitecore.ContentSearch.Analytics">
          <!-- here we name the aggregator -->
          <param desc="name">DefaultObservable</param>
        </processor>
      </aggregation>
      <!-- specific inbound filter for an aggregator -->
      <!-- we use the aggregator name from above in the pipeline name -->
      <defaultobservable.filter.inbound>
        <processor type="Sitecore.ContentSearch.Analytics.Pipelines.AggregationFilters.InboundVisitPathFilter, Sitecore.ContentSearch./
          <filters hint="list:AddFilter">
            <filter>/other.aspx</filter>
          </filters>
        </processor>
      </defaultobservable.filter.inbound>
    </pipelines>
  </sitecore>
</configuration>
```

This configuration uses exactly the same processor types as the global pipeline. Sitecore only applies the processors in the pipeline when the `DefaultObservable` processor is filtering items.

Send feedback about the documentation to docsite@sitecore.net.

The versioned layout API changes

This topic describes the changes to the Sitecore API that the implementation of versioned layouts introduced in Sitecore 8.

The Sitecore.FieldIDs class

The `public static readonly ID FinalLayoutField` field was added to the class. This field stores an identifier of the `__Final Renderings` field.

The Sitecore.Data.Fields.LayoutField class

This section describes a number of changes to the LayoutField class.

The public LayoutField([NotNull] Item item) constructor

Prior to Sitecore 8, the constructor always loaded layout data from the shared `__Renderings` field.

In Sitecore 8 and later, the constructor checks whether the item has a value in the versioned `__Final Renderings` field. If it does, layout data is loaded from this field. If it does not, layout data is loaded from the shared `__Renderings` field.

In Sitecore 8 and later, you should use the constructor to get or to set the layout of an item:

```
var layout = new LayoutField(contextItem).Value;
```

```
new LayoutField(contextItem).Value = layoutXmlValue;
```

The public static string GetFieldValue([NotNull] Field field) method

Prior to Sitecore 8, the public static string GetFieldValue([NotNull] Field field) method only returned layout data from the shared `__Renderings` field. In Sitecore 8 and later, this method can return layout data either from the shared `__Renderings` field or from the versioned `__Final Renderings` field.

The method does not return the raw value of the field. It uses the layout patching workflow to calculate the resulting layout, and it returns this layout.

If you pass the versioned `__Final Renderings` field as an argument, the XML patch from this field is applied to the resulting layout.

If you pass the shared `__Renderings` field as an argument, the value of `__Final Renderings` field is ignored.

In Sitecore 8 and later, you should use the following method to get layout data when you want to ignore the value of the versioned `__Final Renderings` field:

```
var sharedLayout = LayoutField.GetFieldValue(item.Fields[FieldIDs.LayoutField]);
```

The public static void SetFieldValue([NotNull] Field field, [NotNull] string value) method

Before Sitecore 8, you could only use this method to save layout data to the shared `__Renderings` field.

In Sitecore 8 and later, you can use this method to save layout data either to the shared `__Renderings` field or to the versioned `__Final Renderings` field.

The method does not always save the complete layout value to the field. If possible, it saves only the XML patch to the selected field.

In Sitecore 8 and later, you should use the following method to save layout data to the shared `__Renderings` field:

```
LayoutField.SetFieldValue(item.Fields[FieldIDs.LayoutField], sharedLayout);
```

The Sitecore.Data.Items.ItemUtil class

- The `public static void SetLayoutDetails([NotNull] Item item, [NotNull] string sharedLayout, [NotNull] string finalLayout)` method was added.

  In Sitecore 8 and later, you should use this method to save the shared layout and the final layout of an item:

  ```
  ItemUtil.SetLayoutDetails(item, sharedLayout, finalLayout);
  ```

- The `public static void ResetLayoutDetails([NotNull] Item item, bool resetShared, ResetFinalLayoutOptions resetFinal)` method was added.

  In Sitecore 8 and later, you should use this method to reset shared and final layouts of an item:

  ```
  ItemUtil.ResetLayoutDetails(item, resetSharedLayout, resetFinalLayout);
  ```

The Sitecore.Data.Items.RenderingItem class

The `public string GetMultiVariateTestForLanguage([NotNull] Language language)` method was removed.

The Sitecore.Layouts.MultiVariateTestingExtensions class

The `Sitecore.Layouts.MultiVariateTestingExtensions` class was removed because the shared `__Renderings` field is no longer used for storing multivariate test data for all languages. Each version of the item can have different multivariate test data in the versioned `__Final Renderings` field.

The shared __Renderings field

Multivariate test data for all languages is no longer stored in the shared `__Renderings` field. Each version of an item can have different multivariate test data in the versioned `__Final Renderings` field.

The Sitecore.Shell.Applications.ContentEditor.LayoutField class

The following properties were added to the class:

- `public string FieldID`
- `public string ItemLanguage`
- `public string ItemVersion`

Send feedback about the documentation to docsite@sitecore.net.

Versioned layouts

Versioned layouts make it possible to specify different layouts for different versions and languages of the same item.

You use versioned layouts when you want to:

- Have different layouts for different languages.
- Publish a specific version with its own layout for a specific period.

Sitecore uses versioned layouts internally for various content-testing features, for example, to provide cross-language testing.

In the Standard fields of a versioned item, in the Layout section, two fields are for layouts:

- __Renderings – a shared field where you specify the common layout for all languages and versions of the item.
- __Final Renderings – a versioned field where you specify individual layouts for languages and versions of the item.

The layouts are stored as XML in the fields, and the contents of these fields are merged ("patched") together to create the final presentation that visitors see.

The following flow diagram shows in detail how Sitecore resolves which final layout to use for an item:

vo-diagram

Send feedback about the documentation to docsite@sitecore.net.

Apply a campaign code to an online campaign

A campaign activity is a promotion or advertising initiative designed to encourage people to visit a website. Campaigns run for a defined amount of time. You can use campaigns to get visitors to perform certain actions on your website.

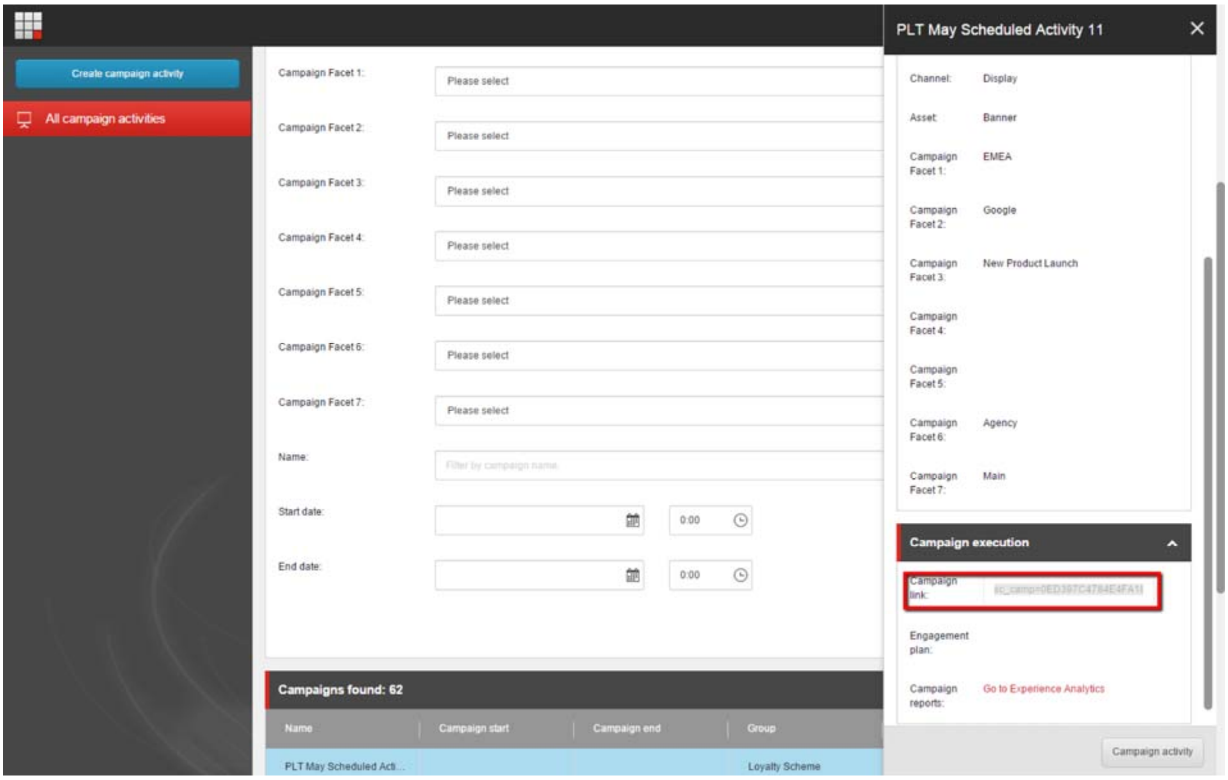
When you create a campaign, the Campaign Creator generates a campaign tracking code. You apply this code to track external interactions, such as email campaigns or traffic coming from an external web source. This enables you to gather more detailed information about campaign traffic and efficiency using the dashboards in Experience Analytics.

Note

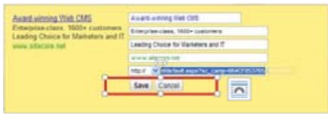
When creating a new campaign activity, you must save the campaign activity before the Campaign Creator generates a campaign code.

To apply a campaign code to an online campaign:

1. From the Sitecore Launchpad, open the Campaign Creator.
2. On the landing page, click the campaign activity you want to associate with the online activity.
3. In the campaign activity panel, under Campaign execution, in the Campaign link field, select the campaign link.



4. Add the campaign link to the website URL where it is appropriate. This may vary from website to website.



5. In this example, the website URL is: www.sitecore.net/default.aspx

   The query string provided for this campaign is:

   sc_camp= 0ED397C4784E4FA1834897A35461992A

   Note

   After you insert the query string, you must insert a question mark to complete the path because this is not included in the Campaign link field.

6. In this example, the new query string is:

   www.sitecore.net/default.aspx?sc_camp= 0ED397C4784E4FA1834897A35461992A

Create an alias for a campaign

To make an alias or a unique URL for your campaign to hide the query string:

1. In the Marketing Control Panel, on the Presentations tab, click Aliases. 
2. Enter the name of your alias and then click Add.
3. When you have finished creating aliases, click Close.

Send feedback about the documentation to docsite@sitecore.net.

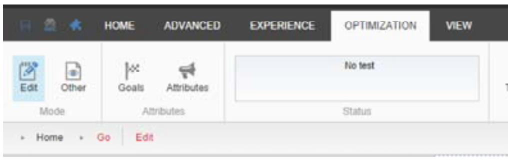
Associate a campaign with an item

You can associate a campaign with various items, such as landing pages or registration forms. By implementing personalization rules, you can customize the items that a viewer sees on your website. You can measure the amount of traffic generated on campaign items and use Sitecore Experience Analytics to evaluate the campaign's effectiveness.

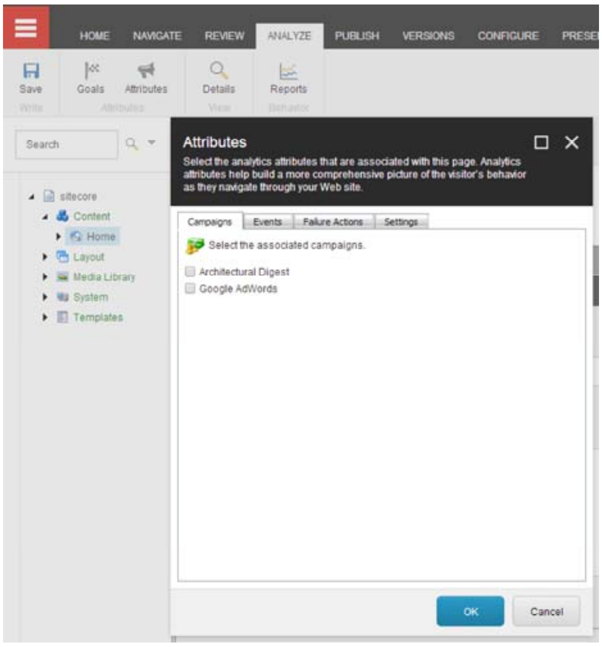
You can associate a campaign with an item in the Experience Editor and in the Content Editor.

To associate a campaign with an item:

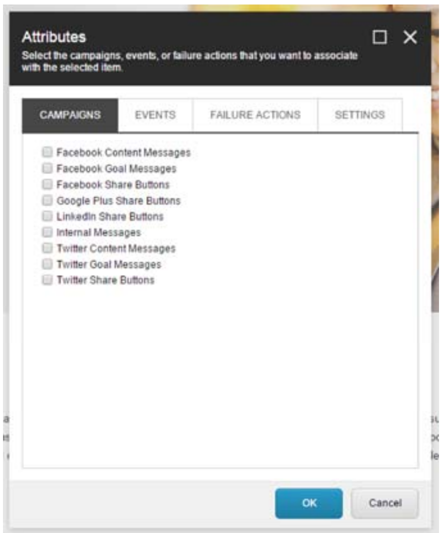
1. Navigate to the item you want to associate with the campaign.
   - In the Experience Editor, navigate to the page that contains the item that you want to associate a campaign with.
   - In the Content Editor, expand the content tree and navigate to the content item that you want to associate a campaign with.
2. In the Attributes group, click Attributes.
   - In the Experience Editor, the Attributes group is found on the Optimization tab.



- In the Content Editor, the Attributes group is located on the Analyze tab.



3. In the Attributes dialog, click the Campaigns tab and then select the campaign that you want to associate with the content item. Click OK.
4. Click Save.



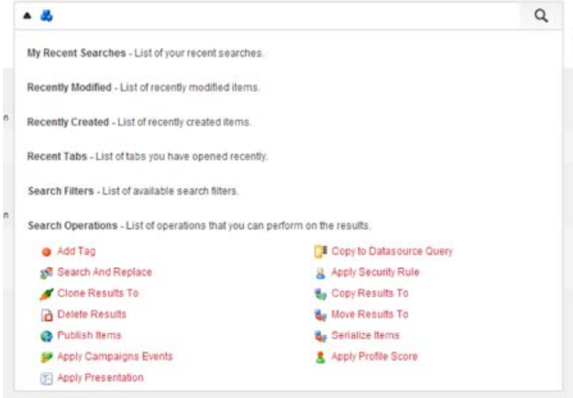
5. On the Home tab, in the Publish group, click Publish to save your changes.

Associate multiple items with a campaign

To associate multiple items with a campaign:

1. Select an item or item bucket.
2. In the search field enter * to return all items in the item bucket.
3. Click the drop-down arrow to the left of the search field and select Search Operations.

4. In Search Operations, select *Apply Campaigns Events.*



5. In the Attributes dialog, select either the event or failure action that you want to associate with the items.
6. Click OK. The items are associated with the even

Send feedback about the documentation to docsite@sitecore.net.

Campaigns

A campaign is a promotion or advertising initiative designed to encourage people to visit your website. Campaigns run for a defined period. You can use campaigns to raise brand awareness or to get contacts to perform certain actions on your website.

In the Sitecore Experience Platform, you can keep track of your campaigns by creating campaign groups and campaign activities. Campaign groups are high-level campaigns that contain multiple campaign activities across different channels. The campaign activities are individual campaigns that run for a limited period on a specific channel.

Identifying campaigns

When you create new campaign activities in the Campaign Creator application, you can identify them by campaign group and channel. These function as the tags that identify which groups the campaign activities belong to. You can create new campaign groups in the Taxonomy node of the Marketing Control Panel. You can also use the Marketing Control Panel to create your own custom campaign facets, which you can use to describe aspects of a campaign that are appropriate to your organization, such as geography, business unit, and so on.

When you create a campaign, the Campaign Creator generates a campaign tracking code. You apply this code to track external interactions, such as email campaigns or traffic coming from an external web source. This enables you to gather more detailed information about campaign traffic and efficiency.

cms_CC_landing

Analyzing campaigns

You can use Experience Analytics to track information about your campaigns. Experience Analytics tracks real-time information about where contacts visiting your website come from, which channels they use to access your content, and which pattern cards they match. You can also view reports about which campaigns are accessed most frequently, and which ones lead to the highest goal conversion rates.

You can use Experience Analytics reports to identify immediate information about the number of contacts visiting your site, as well as the quality – or level of engagement – of the traffic. This helps you evaluate how effective your campaign is and highlight the areas that you can optimize to improve performance.

cms_xAnalytics_campaigns

Personalizing campaigns

You can create personalization rules that let you personalize a campaign's content, making it relevant for each contact. For example, you can track where a contact comes from using GeoIP and use this information in a rule to show different content to contacts, depending on their region or country. You can implement various rules that show personalized content based on the goals a contact has converted, the pattern cards that they match, the number of engagement points that they have accumulated, and so on.

Send feedback about the documentation to docsite@sitecore.net.

Configuring the Campaign Creator

The Campaign Creator is an application that enables marketers to create and organize campaign activities on the Sitecore Experience Platform. The Campaign Creator must be run on Sitecore Experience Platform 8.0 update 2 or higher.

Configuration

The following configuration files are stored in the `\App_Config\Include\Marketing` directory:

- `Sitecore.Marketing.Campaigns.Client.config` registers JavaScript files
- `Sitecore.Marketing.Campaigns.Services.config` registers the application's WebApi controllers

Server roles

In a distributed environment, you must disable the Campaign Creator configuration files on content delivery, reporting, and processing servers.

The Campaign Creator should only be installed on a content management server.

Security roles

To use the Campaign Creator, you must be a member of the following security roles:

- *Analytics Maintaining*
- *Analytics Reporting*

Send feedback about the documentation to docsite@sitecore.net.

Create a campaign group

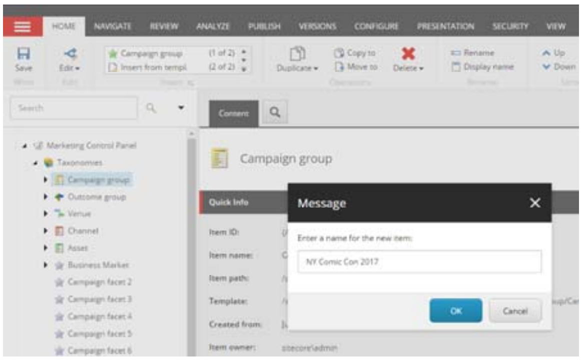
A campaign group is a higher-level campaign that contains multiple campaign activities. You can use campaign groups to group individual campaign activities that you run across different channels.

For example, a campaign group could encompass a collection of activities that promote a marketing event, such as a product launch, seasonal advertising, or a new customer initiative.

When you create a new campaign activity, it is a good idea to associate it with a campaign group. The campaign group functions as a tag to identify campaign activities that belong to the particular campaign group. This is especially useful for creating reports.

To create a campaign group:

1. In the Marketing Control Panel, in the content tree, expand the *Taxonomies* node and click *Campaign Group*.
2. On the Home tab, in the Insert group, click *Campaign Group*.
3. In the Message dialog box, enter a name for the campaign group and click OK.

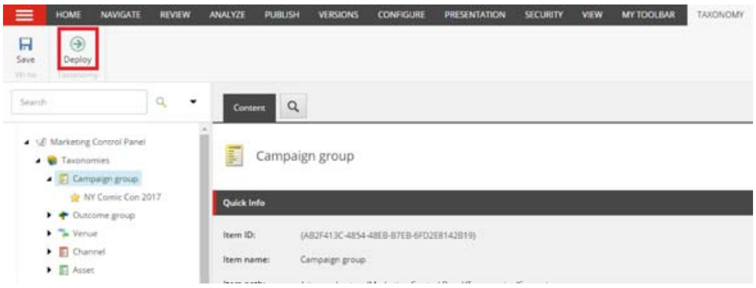


4. To save the new campaign group, click Save.

   Note

   You must deploy the entire *Campaign group* node before you can apply a new campaign group to campaign activities.

5. To deploy the campaign group, select the root *Campaign group* item and on the Taxonomy tab, click Deploy.



Send feedback about the documentation to docsite@sitecore.net.

Create and edit a campaign activity

A campaign activity is a promotion or advertising initiative designed to encourage people to visit your website. Campaign activities run for a defined amount of time. You can use campaigns to encourage visitors to perform certain actions on your website.

After you have created a campaign, you can associate attributes such as campaign groups, channels, and so on, with it. This helps you identify various aspects of the campaign that you can track in Experience Analytics.

This topic outlines how to:

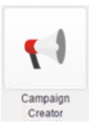
- Create a campaign activity
- Edit an existing campaign activity

Create a campaign activity

Campaign activities are the individual campaigns that run for a defined amount of time.

To create a campaign activity:

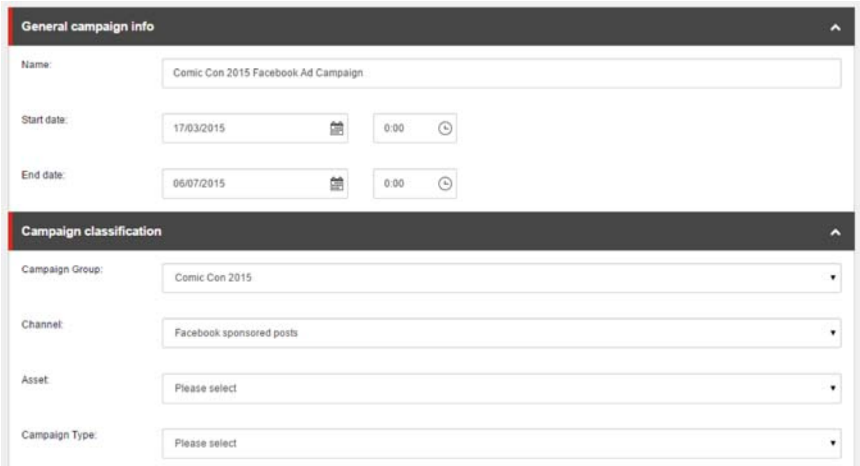
1. On the Sitecore Launchpad, click Campaign Creator.



2. On the landing page, click Create campaign.

3. In the General campaign info section, in the Name field, enter a name for the campaign. To enter start and end dates, click the calendar icon and select a date from the calendar.
4. In the Campaign classification section, select the campaign classification attributes that you want to associate with the campaign activity by clicking the drop-down menu next to each attribute.

   In this example, the *Comic Con 2015* Campaign Group and the *Facebook sponsored posts* Channel have been added to the campaign activity.



5. Click Save to save the campaign activity.

Edit an existing campaign activity

Once you have created a campaign activity, you can edit the campaign's name, start and end dates, or the campaign classification information.

To edit an existing campaign:

1. On the Campaign Creator landing page, click the name of the campaign that you want to edit.
2. In the campaign panel, click *Campaign activity*.
3. On the campaign page, edit the field or fields that you want to change.
4. Click Save to save your changes.

Send feedback about the documentation to docsite@sitecore.net.

Filter and search for a campaign activity

Once you have created a campaign activity in the Campaign Creator, you can use filters to sort reports by different criteria, such as campaign group or channel. This provides you with an overview of how your marketing activities are organized. Once you have created a filtered list of campaign activities, you can open specific items and edit them, or see how they are performing in Experience Analytics.

This topic outlines how to:

- Filter campaign activities
- Search for a campaign activity

Filter campaign activities

To filter your campaign activities:

1. In the Campaign Creator, on the All Campaigns page, in the Campaign classification filters section, open a drop-down menu to find and select a filter that you want to apply to the campaign activities. In this example, the *Comic Con 2015* campaign group has been selected.

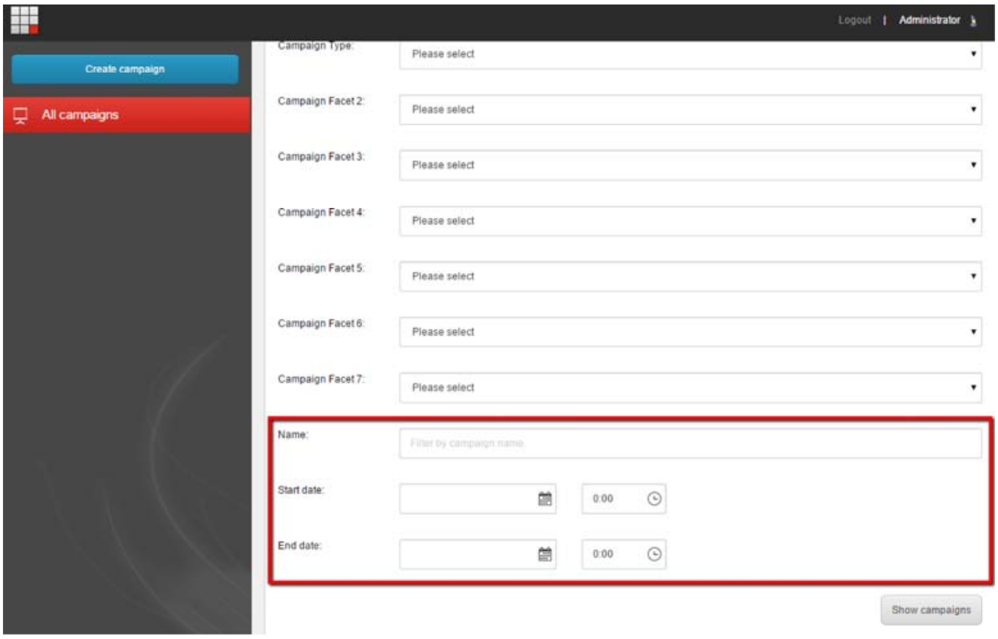
2. Click Show Campaigns to apply your filter settings. You can find all the campaigns that match your filter criteria in the Campaigns found section.

Search for a campaign activity

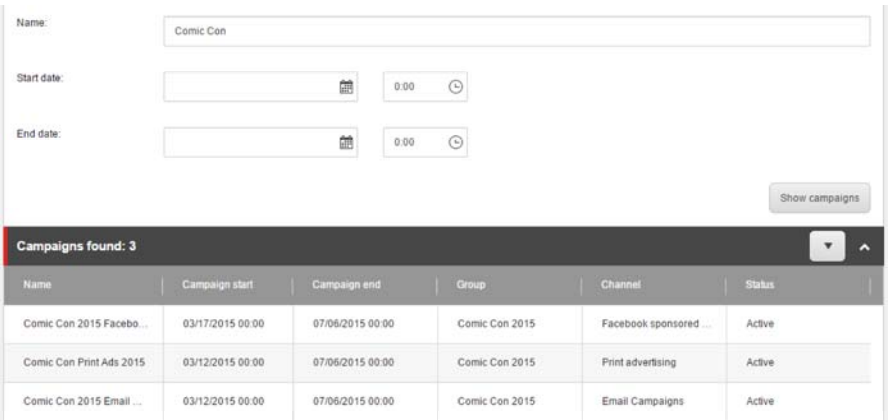
You may want to find specific campaign activities by name, start date, or end date.

To search for a campaign activity:

1. In the Campaign Creator, on the All Campaigns page, in the Campaign classification filters section, in the Name, Start date or End date fields, enter your search criteria, for example, *Comic Con*.



2. Click Show campaigns. Your search results will appear in the Campaigns found section.

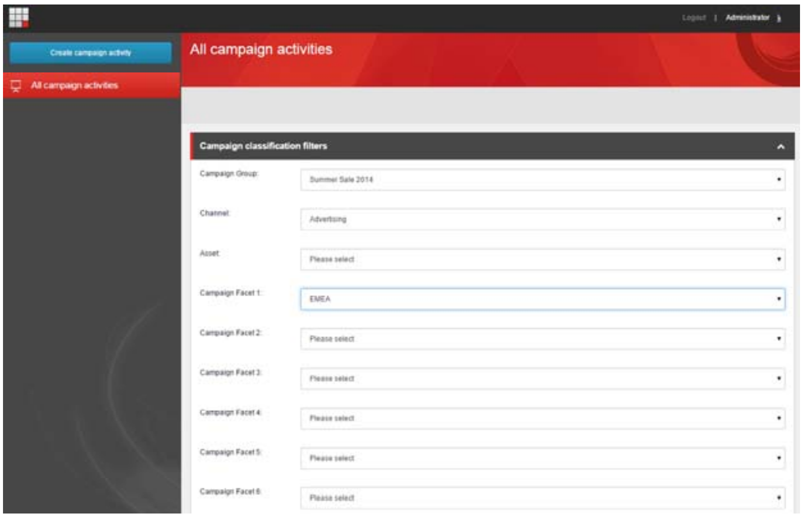


Send feedback about the documentation to docsite@sitecore.net.

The Campaign Creator

The Campaign Creator is a tool that lets you create and organize your campaign activities in one easy-to-use application. You can keep track of all your marketing activities, helping you monitor your campaigns and campaign effectiveness across channels and campaign groups.

You can use the Campaign Creator to create campaign activities. Then, you can organize your campaign activities by channel, campaign group, or other custom facets that reflect your organization's needs. You can create and deploy custom taxonomy tags in the Marketing Control Panel, then use them to organize your campaign activities in the Campaign Creator.

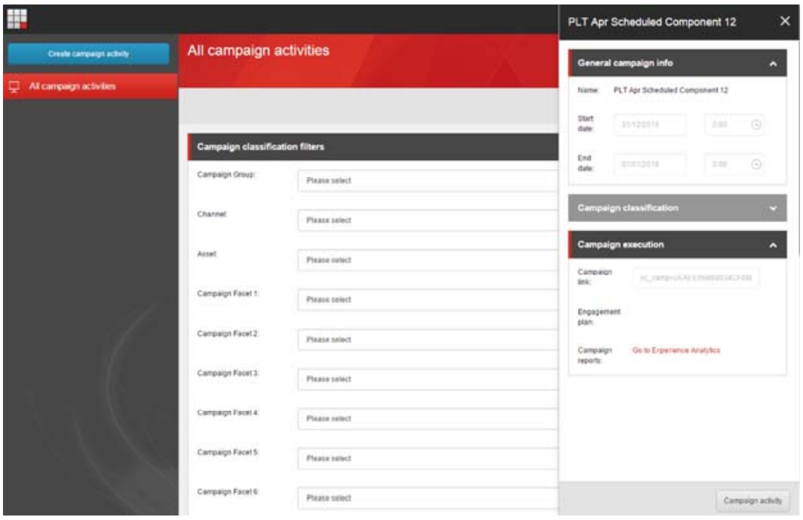


Create campaign activities

When you create a campaign activity, the Campaign Creator generates a campaign tracking code that you can use to track external interactions, such as email campaigns or traffic coming from an external web source. This enables you to gather more detailed information about how effective various channels and campaign activities are at generating traffic and engagement value on your website.

Track campaign activities

Once you have applied the campaign tracking code, you can open Experience Analytics from the Campaign Creator, enabling you to track how your campaigns are performing in real time. You can view traffic levels, engagement value, and engagement value per visit, providing you with information about how effective your campaigns are at generating value and traffic for your website. You can also compare how different campaign activities perform within a single campaign group, giving you insight into which channels provide the most value to your organization, and which channels could use optimization.



Send feedback about the documentation to docsite@sitecore.net.

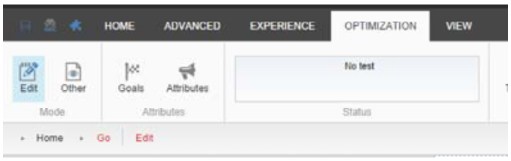
Associate a goal with an item

Goals are activities that visitors can perform on your website. Goals help you track and measure how visitors engage with your website and campaigns. You can associate goals with content, such as landing pages, search fields, and registration forms. You can also associate goals with media items, such as PDF files or videos.

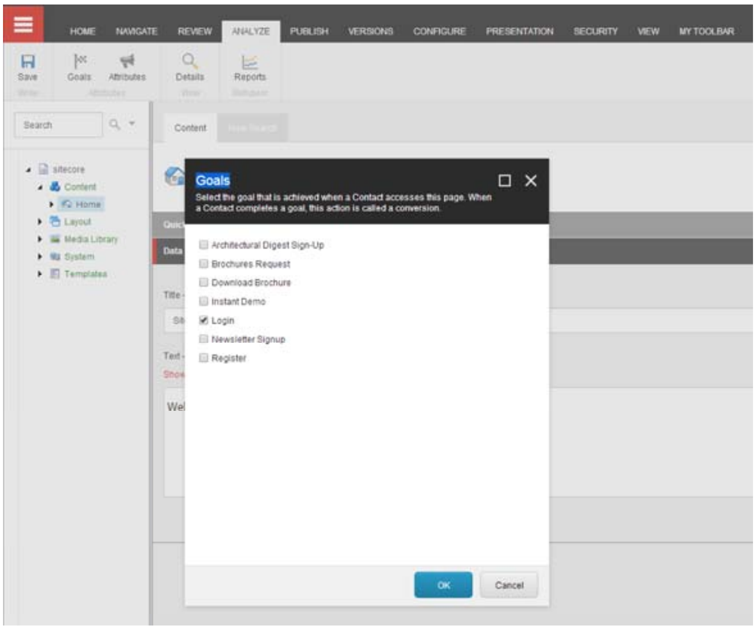
You can associate a goal with a content item in the Experience Editor or in the Content Editor. You use the Media Library or Content Editor to associate a goal with a media item.

To associate a goal with an item:

1. From the Sitecore Launchpad, open the Experience Editor or the Content Editor, then navigate to the relevant content item.
2. In the Attributes group, click Goals.
   - In the Experience Editor, this is located on the Optimization tab

   - In the Content Editor, this is located on the Analyze tab

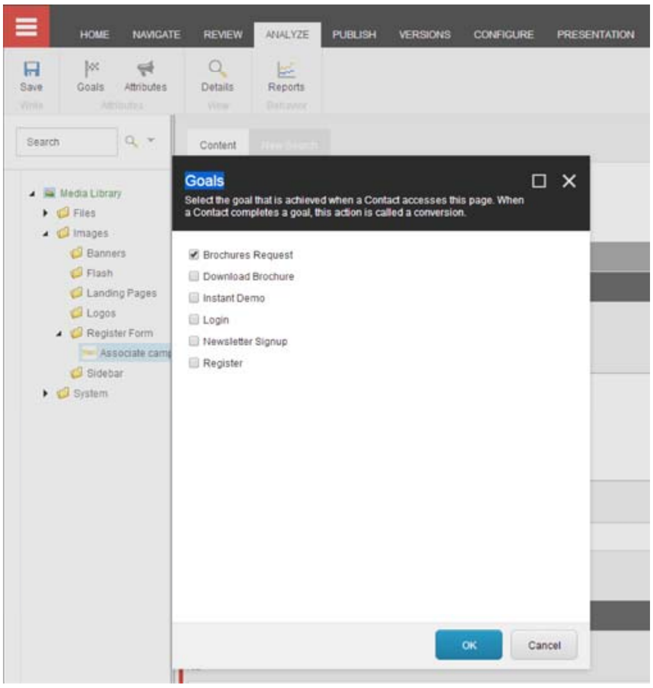
     

3. In the Goals dialog, select the goal that you want to associate with this item and click OK.

Associate a goal with a media item

To associate a goal with a media item:

1. On the Sitecore Launchpad, open the Media Library.
2. Navigate or search for the content item you want to track.
3. On the ribbon, on the Analyze tab, in the Attributes group, click Goals.



4. On the Media Library ribbon, click Save.

Send feedback about the documentation to docsite@sitecore.net.

Associate an event or failure action with an item

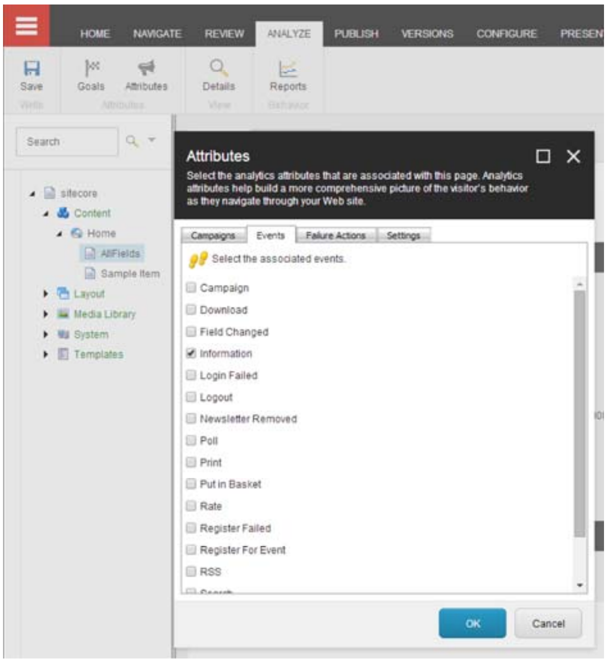
Events track user activity on a website. Failure actions are a type of event that prevent a visitor from completing an action or achieving a goal.

You can associate events and failure actions with items, such as landing pages, registration forms, error pages, or search fields. Tracking events and failure actions help build up a more complete picture of a visitor's behavior as they navigate your website.

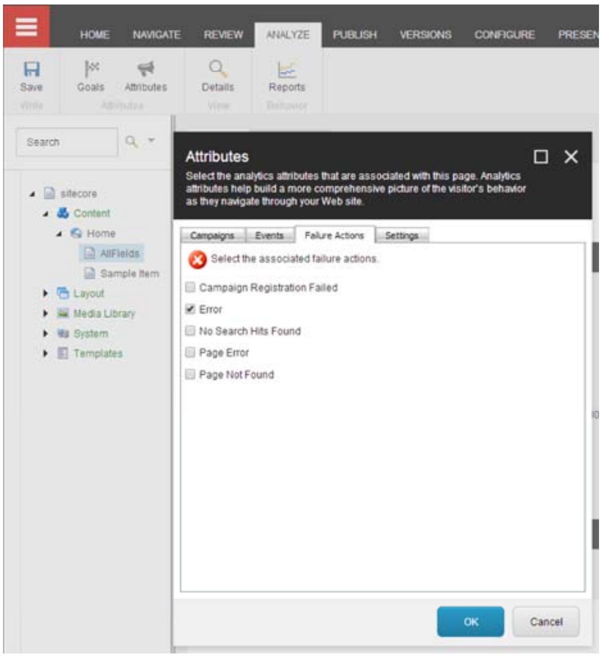
You can also associate multiple items with events or failure actions using the search function in the Content Editor.

To assign an event to a content item:

1. In the Content Editor, expand the content tree and navigate to the item that you want to associate an event with.
2. On the Analyze tab, in the Attributes group, click Attributes. This opens the Attributes dialog box.
3. In the Attributes dialog box:
   - To associate an item with an event, click the Events tab. Select the event that you want to associate with the item.



   - To associate the item with a failure action, click the Failure Actions tab. Select the event that you want to associate with the item.



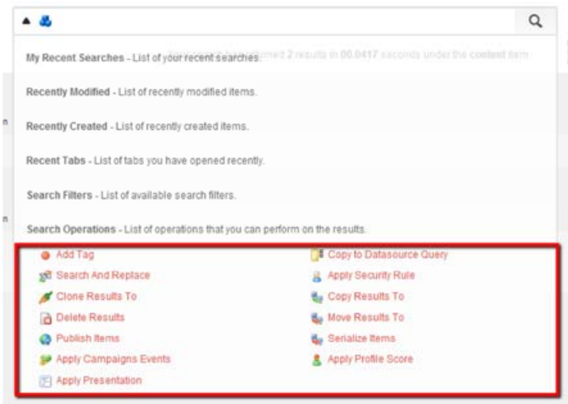
4. Click OK to save your changes and close the Attributes dialog box.

Associate an event or failure action with multiple items

To assign a custom profile value to multiple items:

1. In the Content Editor, select an item or item bucket.
2. In the search field enter * to return all items in the item bucket.
3. Click the drop-down arrow to the left of the search field and select Search Operations.

4. In Search Operations, click Apply Campaign Events.



5. In the Attributes dialog box, select either the event or the failure action that you want to associate with the items.
6. To associate the event with these items, click OK.

Send feedback about the documentation to docsite@sitecore.net.

Create and deploy a goal

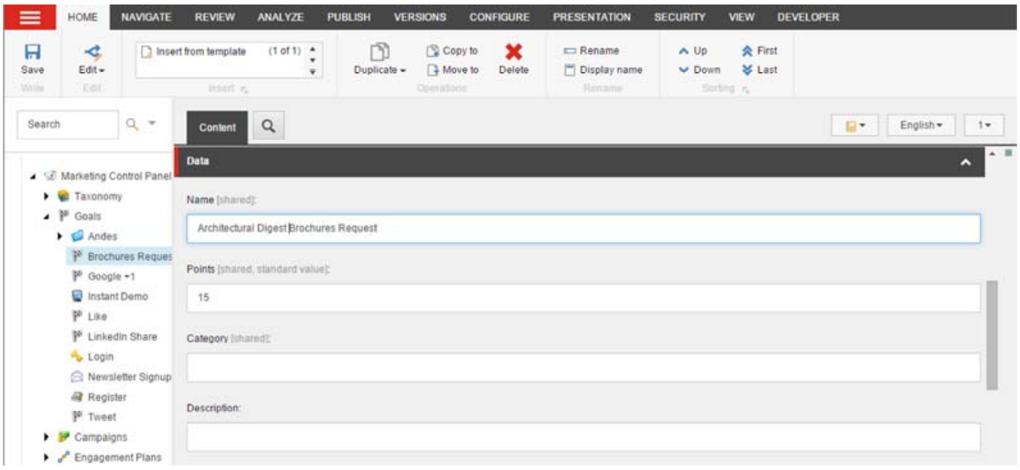
Goals are activities that contacts can perform on your website. You can associate goals with campaign activities and events and then use them to track how contacts engage with your website.

When you create a goal, it is subject to a workflow. A goal must be in the final workflow state to be deployed and published to your website. You must deploy the new goal before it becomes available in the Marketing Control Panel and you can associate it with a content item or a campaign.

To create a goal:

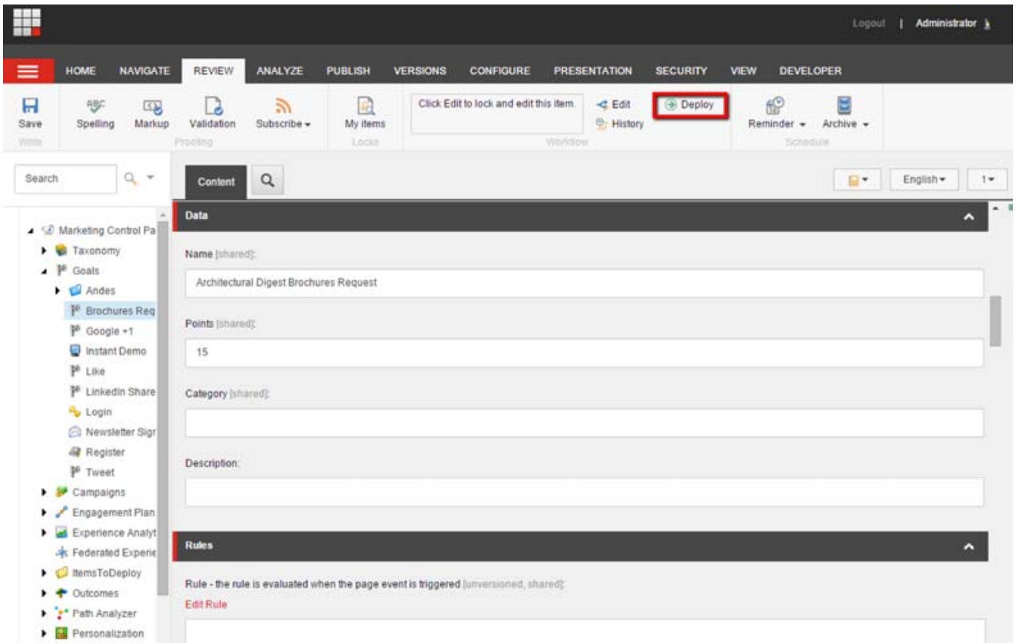
1. In Marketing Control Panel, in the content tree, click *Goals*.
2. On the Home tab, in the Insert group, click *Goal* to create a new goal.
3. In the dialog box that appears, enter a name for the goal.

   On the Content tab, you can enter more detailed information about the goal, such as engagement value points or a description.



Field	Value
Name	Enter the name of your goal
Points	Enter the number of engagement value points that you want to assign to a contact when they achieve this goal.
Category	This field defines the event as a system event. For administrators and advanced users only.
Description	Enter a clear description of the goal.
Rule	Select the rule that should be evaluated when the page event associated with this goal is triggered.
IsAuthorFeedback	Select this checkbox if the page events associated with this goal are of interest to content authors.
IsFailure	Select this checkbox to define the event as a failure. For administrators and advanced users only.
IsGoal	Select this checkbox to ensure that the goal appears in the Content Editor, Goals dialog. You can then associate the goal with a content item.
IsSystem	Select this checkbox if the page event describes a system event.

4. Click Save.
5. To deploy the goal, on the Review tab, in the Workflow group, click *Deploy*.



Now you can associate the goal with content items and campaigns.

Send feedback about the documentation to docsite@sitecore.net.

Events

Events track visitor activity on a website. Tracking events helps build up a more complete picture of a visitor's behavior as they navigate your website. You should assign [engagement value points]() to all events to reflect their relative importance to your organization.

You can track events such as:

- Downloads
- Searches
- Registration
- Logout

Visitors can trigger events in goals and campaigns when they achieve certain objectives on your website. You can also use events to track how visitors come to your site, for example, from promotions or campaigns. The Experience Analytics application creates reports where you can view campaign and conversion-tracking data.

Sitecore administrators can edit existing events and create additional events if you need them. You must remember to specify how many engagement value points you want to assign to each event.

You can also create [engagement plans]() that use events and triggers to select the relevant automated actions for specific stages in your engagement plan.

Send feedback about the documentation to docsite@sitecore.net.

Failure Actions

A failure action is a type of event that prevents a visitor from completing an action or achieving a goal. It can be the result of a technical error or visitor action within the system. You can track failure actions in Sitecore Analytics.

Examples of failure actions that result from technical errors include:

- Displaying an error page when a server is down
- Showing a "no search hits found" message after an unsuccessful query

Examples of failure actions that result from visitor actions include:

- Displaying an error message after a visitor enters incorrect login information
- Displaying an error message after a visitor enters an invalid email address

You can associate failure actions with single items, or [use search operations]() to assign failure actions to multiple items.

Send feedback about the documentation to docsite@sitecore.net.

Goals

Goals are activities that visitors can perform on your website. You create goals to track and measure how visitors engage with the website and campaigns – both online and offline.

Some examples of goals:

- Download a brochure
- Register for an email newsletter
- Visit a particular page
- Sign up for an online demo

Once you have created some goals, you can measure the conversion rate for each goal – the percentage of visitors that achieve a particular goal. You can find charts and dashboards that show goals and conversion rates in Experience Analytics.

cms_xAnalytics_goals

Using goals

You can use goals and conversion rates to measure visitor engagement. For example, you could create an engagement plan that uses goals to evaluate which actions should be taken at different stages of the engagement plan.

You can also use goals to create personalization rules that are based on the goals that a visitor achieves during a visit. If a visitor has signed up for a newsletter, for example, you could create a personalization rule which then shows this visitor content instead of the newsletter sign-up item.

Goal facets

In the Marketing Control Panel, under marketing taxonomies, you can use goal facets to describe different aspects of a goal that you might want to track. These could be categories for your goals or descriptions that you can apply to multiple goals in order to better organize them.

You can use goal facets to create and implement personalization rules that show content to your contacts based on the type of goal that they have triggered. You can also create custom reports that track the goal facets, providing you with deeper insights into how contacts use and interact with your websites.

Actions and rules in engagement plans

An engagement plan is a predefined plan that lets you control how your website interacts with some visitors. When you create an engagement plan, you create a system whereby a particular part of certain visitors' online activity is subject to a set of rules and conditions. These rules and conditions determine when specific actions are performed for these website visitors.

Predefined actions and rules

The Sitecore Marketing Center comes with a number of predefined actions and rules that you can use to create engagement plans.

You can use a predefined action to, for example:

- Increase or decrease the profile value of a visitor
- Add or remove the visitor from an engagement plan or state
- Send an e-mail message

In engagement plans, you can use the predefined rules to evaluate information, such as:

- The level of authentication that the visitor has.
- The country, city, postal code, and IP address that the visitor comes from.
- The number of times the visitor has been on the website.
- Whether the visitor has triggered a campaign event.

Additional actions and rules

Engagement plans can vary in size and scope, depending on any number of factors, including the business you are in, the size of your website, your target audience, what kind of campaigns you are running, and so on.

If the predefined actions and rules do not let you create the engagement plan that you need for your website, your IT department or Sitecore partner can easily create additional conditions, rules, and actions for you, as well as any pages, forms, security roles, and so on that you may need.

Send feedback about the documentation to docsite@sitecore.net.

Engagement plan triggers and conditions

Triggers are the events that compel Sitecore to evaluate the conditions that you define in rules within engagement plans. Conditions are a series of rules that determine whether visitors in a specific state fulfill certain criteria. You use triggers and conditions to move visitors to and from different states in engagement plans.

In engagement plans, there are two types of triggers:

- Visitor triggers
- Timeout triggers

Visitor triggers

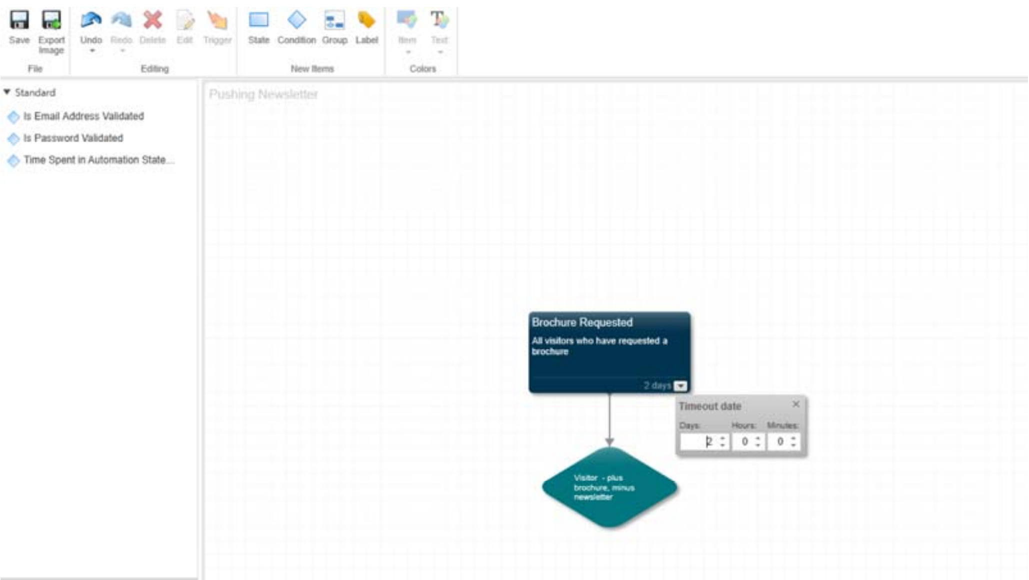
A visitor trigger specifies that when a visitor achieves a page event or goal, Sitecore automatically evaluates the rule to determine whether or not the visitor meets that condition and what action should be taken.

For example, you could link a visitor trigger to an event, such as a brochure download. Once a visitor downloads the brochure, they meet a condition based on the event and are moved to the next state in the engagement plan.

Timeout triggers

You use timeout triggers to specify that if a visitor has been in the associated state for a specific time interval, Sitecore must evaluate the linked rule to determine if the visitor meets the condition and what actions should be taken. You can also create timeout triggers that evaluate whether a timeout interval has elapsed since the last time Sitecore evaluated a condition.

For example, you could set up a timeout trigger on an engagement plan state that will evaluate a rule after two days. After that time has elapsed, Sitecore checks to see if a visitor met the conditions in the state, and then moves the visitor on in the engagement plan accordingly.



When you add a condition to an engagement plan, you can specify a visitor trigger, a timeout trigger, or both.

Send feedback about the documentation to docsite@sitecore.net.

Engagement plans

An engagement plan is a predefined plan that lets you control how your website interacts with some visitors. You use engagement plans to nurture relationships with your visitors by adapting communication to use the content, channels, and media that are appropriate for each individual interaction.

When you create an engagement plan, you create a system whereby a particular part of certain visitors' online activity is subject to a set of rules and conditions that determine when specific actions are performed for these website visitors.

An engagement plan consists of:

- States – the status of a website visitor, such as whether or not they have visited certain pages, achieved specific goals, performed specific actions, where they have come from, and so on.
- Conditions – logic-based statements that you set which determine whether visitors in a specific state fulfill certain criteria. Conditions can relate to the visitor in general or this particular visit.
- Rules – criteria that you define which determine whether a visitor fulfills a condition.
- Triggers – events that determine when Sitecore should evaluate the conditions and rules associated with a state.
- Actions – you set actions that should be taken when visitors meet or fail to meet the rules specified in a condition.

Planning an engagement plan

Before you can create an engagement plan, you should ensure that all of the elements that you need to make the plan function are in place. This could include specific pages, graphical elements, email messages, campaigns, and so on. You should also have a logical scenario that you want to implement.

Once you have all the elements in place, you can design an engagement plan that implements your scenario and seamlessly makes specific actions take place when visitors meet the conditions you specify.

Send feedback about the documentation to docsite@sitecore.net.

Events

Events track visitor activity on a website. Tracking events helps build up a more complete picture of a visitor's behavior as they navigate your website. You should assign engagement value points to all events to reflect their relative importance to your organization.

You can track events such as:

- Downloads
- Searches
- Registration
- Logout

Visitors can trigger events in goals and campaigns when they achieve certain objectives on your website. You can also use events to track how visitors come to your site, for example, from promotions or campaigns. The Experience Analytics application creates reports where you can view campaign and conversion-tracking data.

Sitecore administrators can edit existing events and create additional events if you need them. You must remember to specify how many engagement value points you want to assign to each event.

You can also create engagement plans that use events and triggers to select automated actions for specific stages in your engagement plan.

Send feedback about the documentation to docsite@sitecore.net.

Failure Actions

Failure actions are a type of event that prevent a visitor from completing an action or achieving a goal. They can be the result of technical errors or visitor actions within the system. You can track failure actions in Sitecore Analytics.

Examples of technical errors that result in a failure action include:

- Displaying an error page when a server is down
- Showing a "no search hits found" message after an unsuccessful query
- Examples of visitor actions that result in a failure action include:
- Entering incorrect login information
- Entering an invalid email address

You can associate failure actions with single items, or use search operations to assign failure actions to multiple items.

Send feedback about the documentation to docsite@sitecore.net.

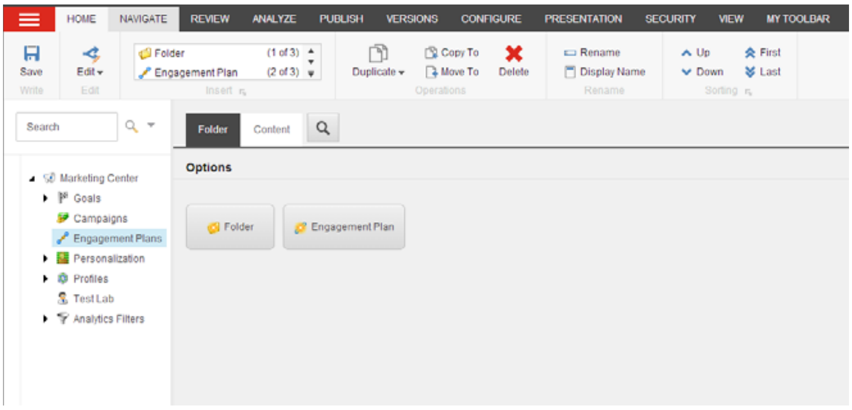
Add a label to an engagement plan

An engagement plan is a predefined plan that lets you control how your website interacts with some visitors. You use engagement plans to nurture relationships with your visitors by adapting communication to use the content, channels, and media that are appropriate for each individual interaction.

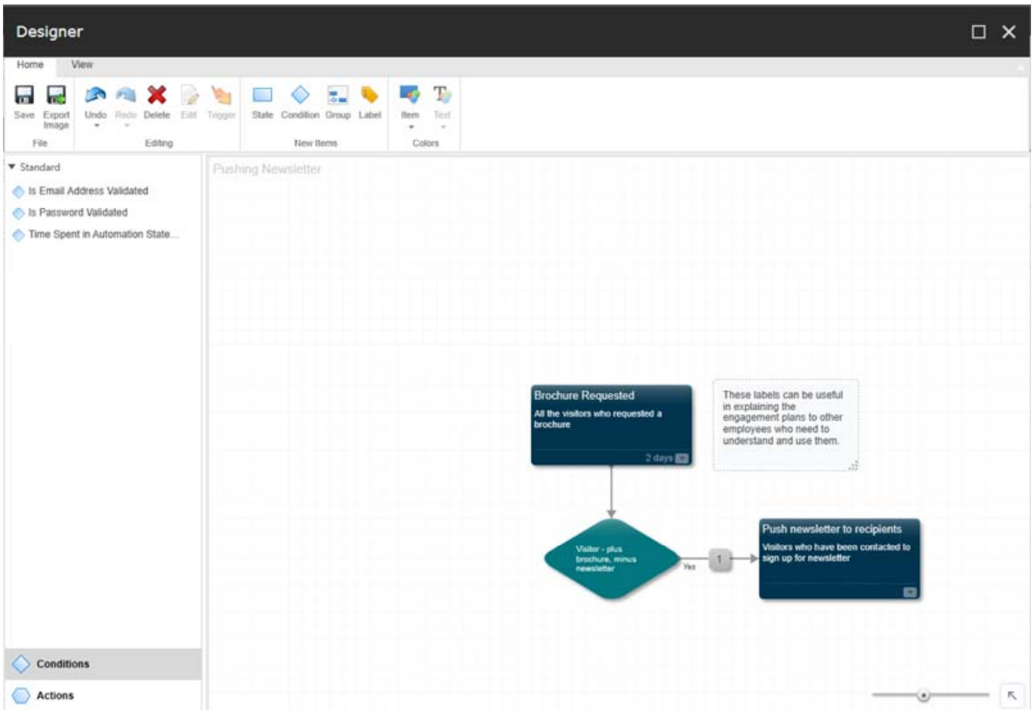
As your engagement plans grow and get more complicated, you may need to add labels that contain descriptions of the plan and the situations that it is designed to address. These labels can be useful in explaining the engagement plans to other employees who need to understand and use them.

To add a label:

1. In the Marketing Control Panel, in the content tree, click Engagement Plans.



2. Select your engagement plan, then, on the Engagement Plan tab, in the Options section, click Design to open the Designer.
3. In the New Items group, click Label and a new item appears in the engagement plan.
4. Select the new item and drag it to an appropriate place.
5. In the new item, double-click the text and enter the new text that you want to insert into the engagement plan.
6. Resize the label and drag it to an appropriate place in the plan.



Send feedback about the documentation to docsite@sitecore.net.

Add a segment to a state in an engagement plan

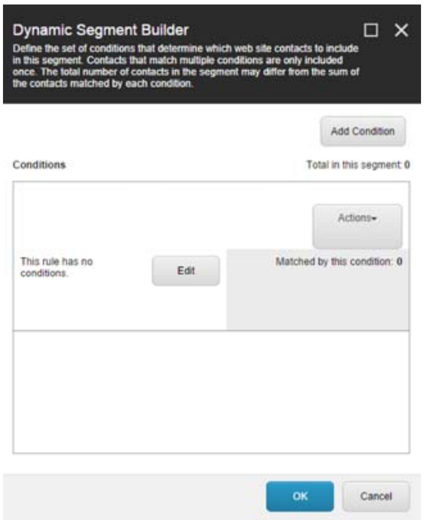
Within engagement plans, you can create segments that target specific groups of contacts in a state. You can use segments to define engagement plan actions that allow you to send relevant content to the contacts within that segment.

You can segment contacts in an engagement plan by using the Dynamic Segment Builder – a tool that lets you specify a list of conditions and rules that determine which contacts belong in the segment. Only contacts who meet the specified conditions and rules are added to the segment. You can use combinations of several conditions and rules to create more complex segments.

For example, you can create a segment, to target all engagement plan contacts in a certain state who are from a particular region or country. You can then send these contacts an e-mail containing content relevant to their interests.

To create a segment of contacts:

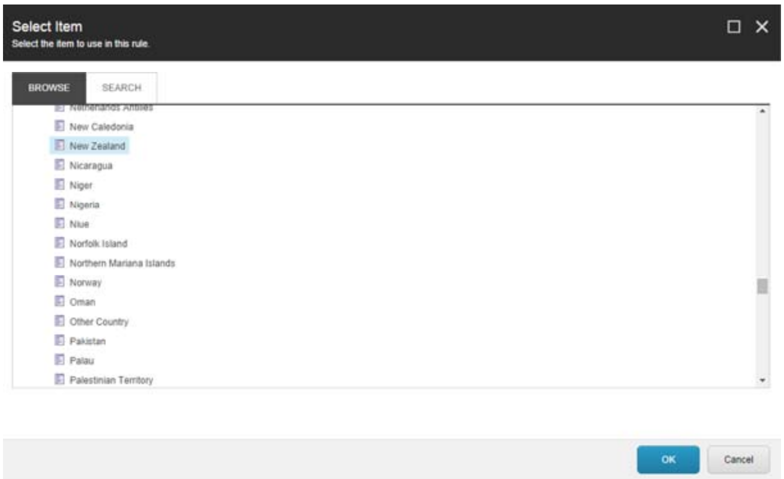
1. In the Supervisor, select the state that you want to add the segment to.
2. On the Home tab, in the Contacts group, click Add, and then click Add a Segment.
3. In the Dynamic Segment Builder dialog box, click Add Condition.



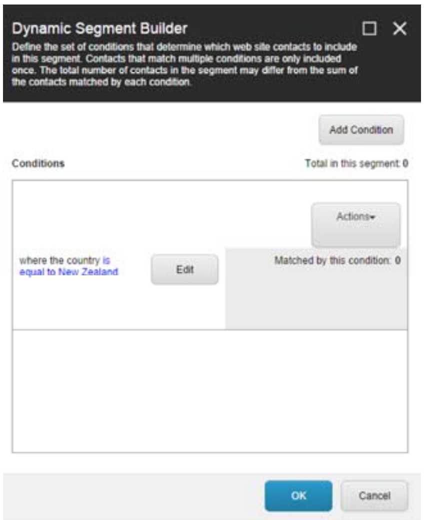
4. Click Edit.
5. In the Rule Set Editor dialog box, in the Select the conditions for the rule field, select a condition that the contacts must fulfill to be in this segment.

   For example, you can implement a rule that contacts in the segment should be from a particular country or region.

6. In the Rule description field, click *compares to,* and then in the Select Comparison dialog box, select the comparison that you want to use for this rule.
7. For example, selecting *is equal to* from the Select Comparison dialog box would allow you to target contacts from a specific country.
8. In the Rule description field, click the specific item or number to select the last part of your condition.
9. In the Select Item dialog box, make a selection and then click OK.



10. In the Rule Set Editor dialog box, click OK and the condition is added to the Dynamic Segment Builder dialog box.



The Matched by this condition field tells you how many contacts fulfill this condition.

The Total in this segment field tells you how many contacts are in the current segment.

11. Click OK.

    The users in this segment are now added to the current state in the engagement plan.

Send feedback about the documentation to docsite@sitecore.net.

Add a visitor to an engagement plan

Engagement plans are predefined plans that let you control how your website interacts with some visitors. You can enroll visitors in engagement plans by participating in campaigns, by filling out web forms, or by adding them directly to the engagement plan.

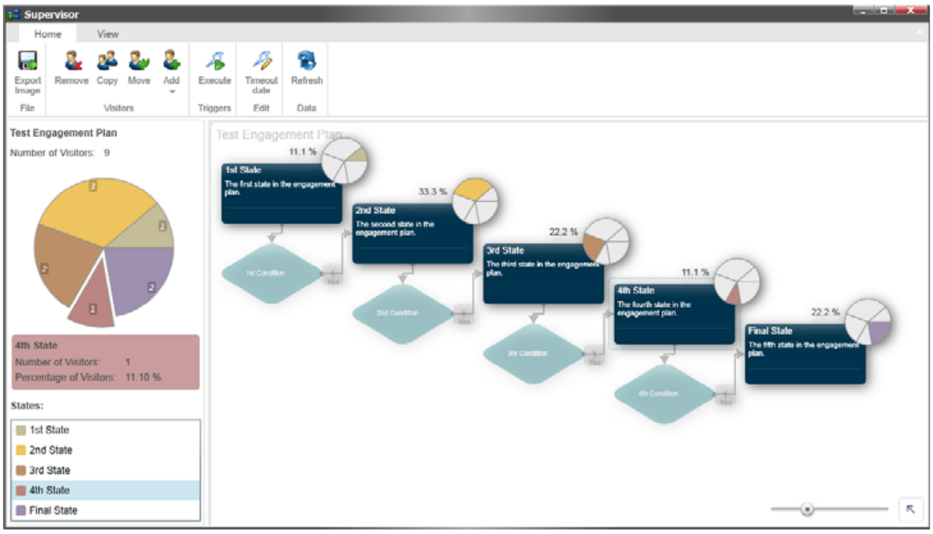
This topic outlines how to manually add visitors to a state in an engagement plan using the Supervisor. You can also use this functionality to test if your engagement plan works as expected.

Note

To add visitors automatically to an engagement plan, you can use a web form or campaigns.

To manually add a visitor to an engagement plan:

1. In the Marketing Center, select the engagement plan that you are interested in and then click Supervise.
2. In the engagement plan, select the state that you want to add a visitor to. The supervisor highlights that slice of the pie and tells you how many visitors are in that state.



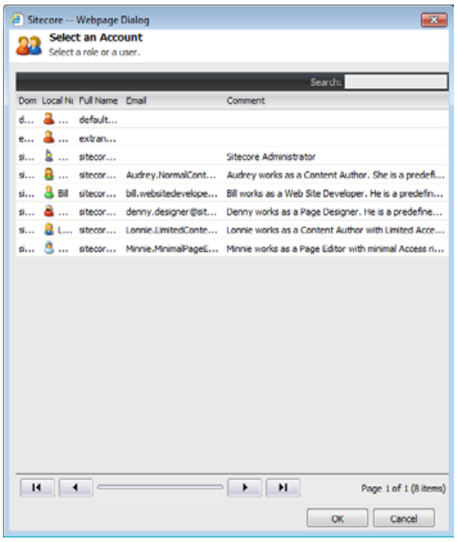
3. On the ribbon, on the Home tab, in the Visitors group, click Add.

   There are four ways to add visitors to a state:

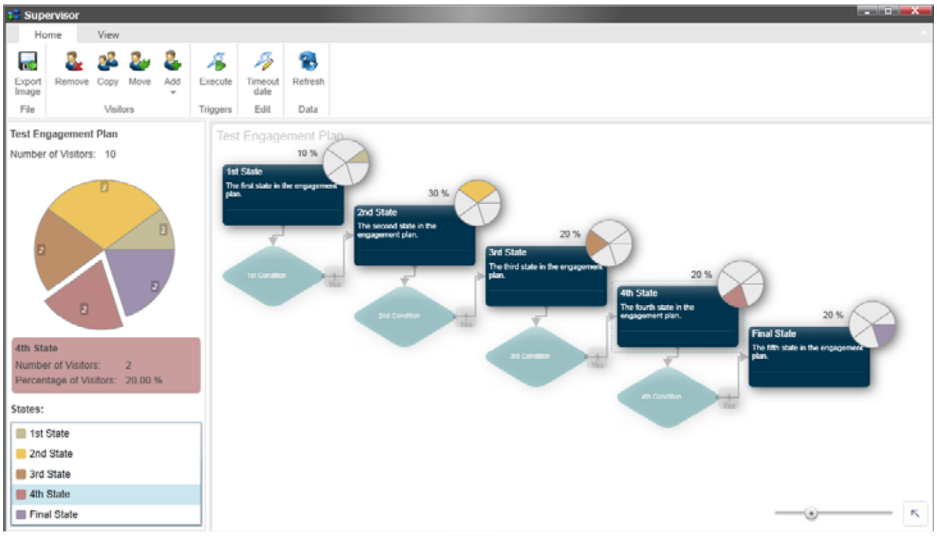
Option	Description
Add From CSV File	You can add all the visitors that are listed in a comma separated value file.

Add a Sitecore User	You can add a specific user from the list of visitors who are in the Sitecore security system.
Add all Members of a Sitecore Role	You can add all the visitors who are members of a particular Sitecore role.
Add a Segment	You can create a segment of the visitors who are in the Sitecore security system and add it to the state.

4. Click Add a Sitecore User and the Select an Account dialog box appears.
5. In the Select an Account dialog box, select the user that you want to add to this state in the engagement plan and click



6. In the Supervisor, you can see that the user is added to the current state and the size of the corresponding slice has increased. You may need to click Refresh to update the data.



Send feedback about the documentation to docsite@sitecore.net.

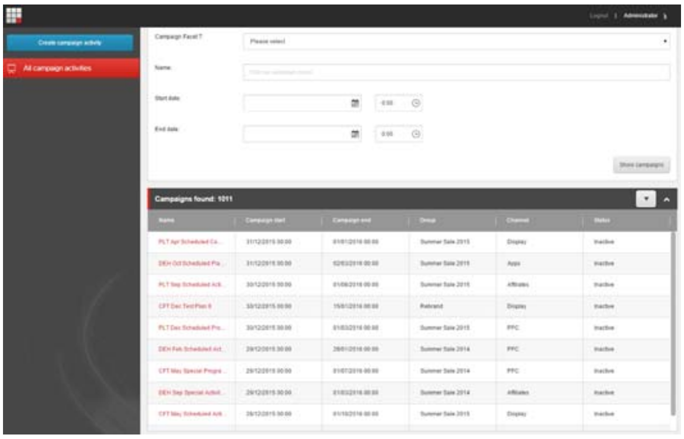
Associate a campaign activity with an engagement plan

Engagement plans are predefined plans that let you control how your website interacts with some visitors. You can enroll visitors in engagement plans by participating in campaigns, by filling out web forms, or by adding them directly to the engagement plan. This topic explains how to use a campaign to enroll visitors in an engagement plan.

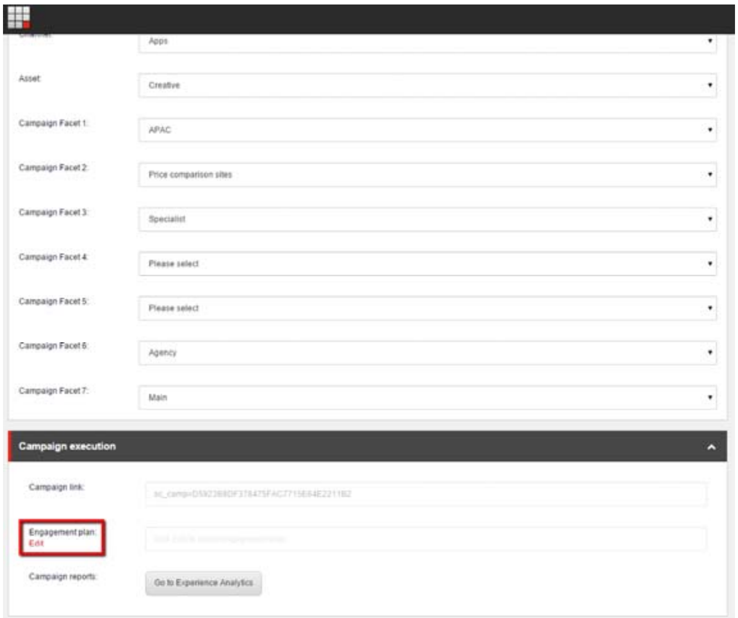
A campaign activity is a promotion or advertising initiative designed to encourage people to visit a website, which runs for a defined amount of time. When you create a campaign activity, you can specify that visitors who participate in a campaign are automatically enrolled in an engagement plan.

To link a campaign activity to an engagement plan:

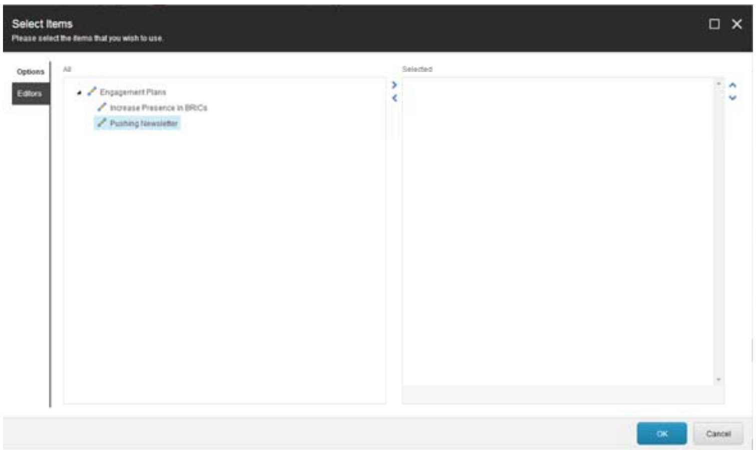
1. In the Campaign Creator, click the campaign activity that you want to link to an engagement plan.



2. In the campaign activity, in the Campaign execution section, under the Engagement plan field, click Edit. The Select Items dialog box appears.



3. In the Select Items dialog box, expand the engagement plan that you are interested in, for example, *Pushing Newsletter*, and select the state that you want to enroll visitors in when they participate in this campaign.



4. Click OK.

Send feedback about the documentation to docsite@sitecore.net.

Execute an engagement plan trigger

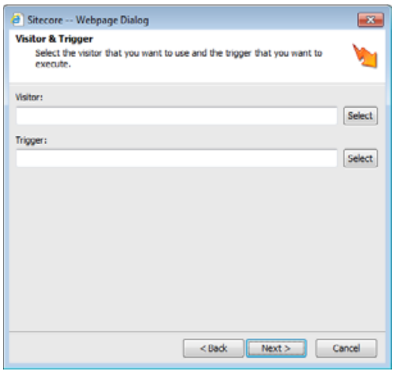
An engagement plan is a predefined plan that lets you control how your website interacts with some visitors. You use engagement plans to nurture your relationships with your visitors by adapting the communication to use the content, channels, and media that are appropriate for each individual interaction.

Triggers are the events that compel Sitecore to evaluate the conditions that you define in the rules in engagement plans. You use triggers and conditions to move visitors to and from different states in the engagement plans.

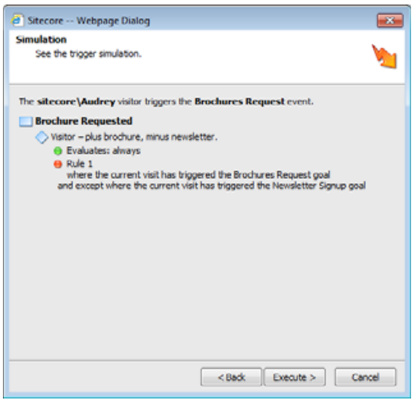
You can execute the trigger associated with a state in an engagement plan and force the plan to evaluate the conditions associated with that particular state. You do this to move a specific visitor to a different engagement plan state.

To execute a trigger:

1. In the Supervisor, select a state that contains the visitor.
2. In the Triggers group, click Execute.
3. In the Execute Trigger Wizard wizard, click Next.



4. In the Visitor & Trigger window, in the Visitor field, click select and click the visitor that you want to force the trigger for.
5. In the Trigger field, click Select and click the Trigger that you want to force for the visitor.



The wizard displays an overview of the user, the trigger that you selected, and the rules that it evaluates.

6. Click Execute to execute the trigger. If the visitor meets the conditions specified for the trigger, the actions that you have defined are performed for that visitor

Send feedback about the documentation to docsite@sitecore.net.

Move, copy, and delete engagement plan visitors

An engagement plan is a predefined plan that lets you control how your website interacts with certain visitors. After you have deployed an engagement plan, you can manage details about the visitors enrolled in the plan from the Supervisor.

In the Supervisor, you can:

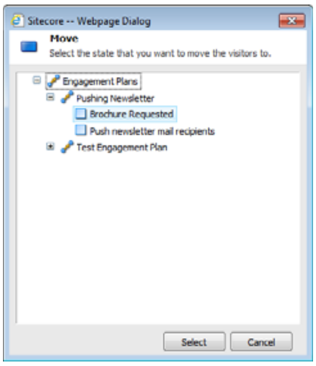
- Move visitors from one engagement plan to another
- Copy visitors from one engagement plan to another
- Delete visitors from an engagement plan

Moving visitors to another plan

You can move visitors from one engagement plan to another engagement plan. For example, if an engagement plan is not working as intended, you can move all visitors enrolled in the engagement plan to another plan.

To move a visitor:

1. In the Marketing Control Panel, select the relevant engagement plan and open the Supervisor.
2. In the Supervisor, select the state whose visitors you want to move.
3. In the Visitors group, click Move.
4. In the Move dialog box, select the engagement plan and the state that you want to move the visitors to.



When you move visitors to another state in the same engagement plan, they are removed from the state that they are currently in and placed in the state that you select. You can also move visitors from one state to another within the same plan by executing an engagement plan trigger. When you move visitors to another state in another engagement plan, they are removed from the state in the current engagement plan and placed in the state you select in the other engagement plan.

Note

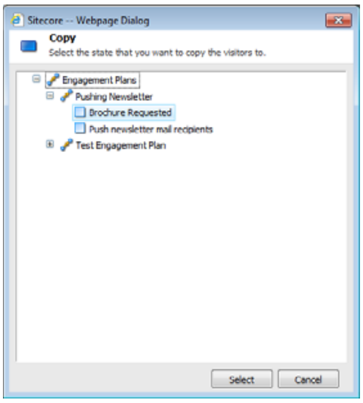
A visitor can only be in one state in an engagement plan.

Copying a visitor

You can also copy a visitor from one engagement plan to another. For example, you can do this to test a newly created engagement plan.

To copy a visitor:

1. In the Marketing Center, select the relevant engagement plan and open the Supervisor.
2. In the Supervisor, select the state that contains the visitors that you want to copy.
3. In the Visitors group, click Copy.
4. In the Copy dialog box, expand the engagement plan that you want to copy the visitors to and then select the state that you want to move them to.



You cannot copy visitors to a state in the same engagement plan. A visitor can only be in one state in an engagement plan.

When you copy visitors to a state in another engagement plan, they are not removed from the original state.

Deleting a visitor from an engagement plan

To delete visitors from an engagement plan:

1. In the Marketing Center, select the relevant engagement plan and open the Supervisor.
2. In the Supervisor, select the state whose visitors you want to delete.
3. In the Visitors group, click Delete.

Send feedback about the documentation to docsite@sitecore.net.

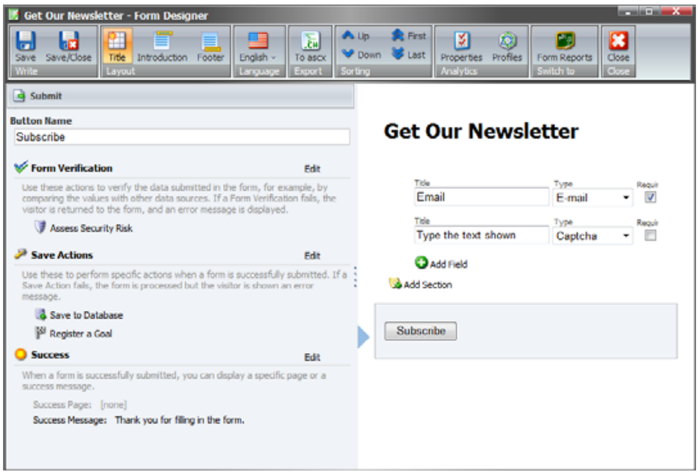
Use a web form to enroll a visitor in an engagement plan

Engagement plans are predefined plans that let you control how your website interacts with some visitors. You can enroll visitors in engagement plans when they participate in campaigns or fill out web forms, or by adding them directly to the engagement plan. This topic outlines how to use a web form to enroll visitors in an engagement plan.

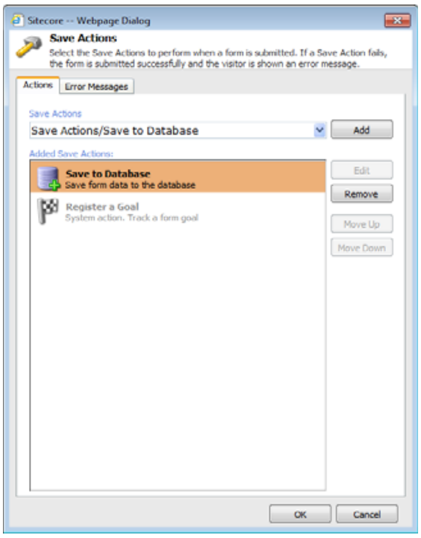
If you use the Web Forms for Marketers module to create the forms that you use on your website, you can specify that visitors who submit a web form are automatically enrolled in an engagement plan. Web Forms for Marketers allows you to create simple forms and localize them in many different languages. The module then records and reports the information provided by visitors in the forms.

To link a form to an engagement plan:

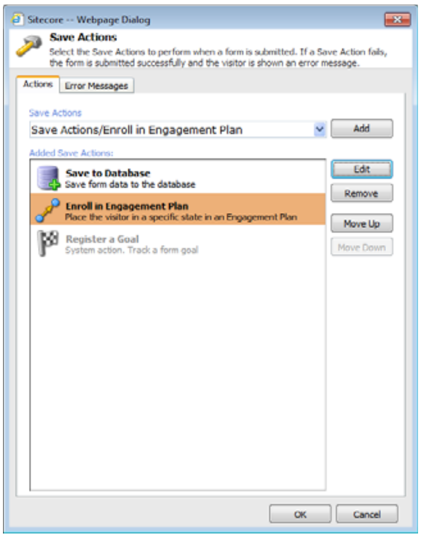
1. In the Sitecore Desktop, click Sitecore, Web Forms for Marketers, Form Designer.
2. In the Form Designer, select the form that you want to link to an engagement plan. It opens in the Form Designer.



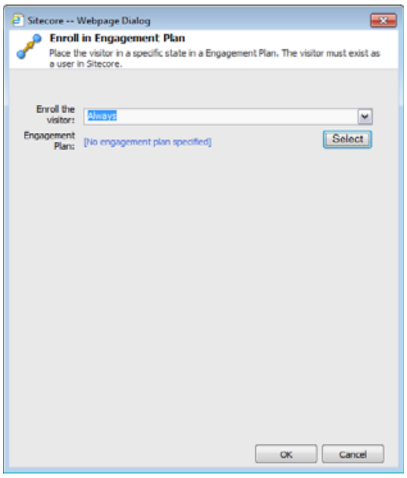
3. In the Form Designer, select the button that the user clicks to submit the form and in the left-hand panel, in the Save Actions section, click Edit and the Save Actions dialog box appears.



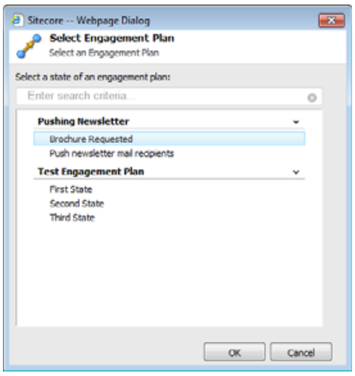
4. In the Save Actions dialog box, in the Save Actions field, click the drop-down arrow and select *Enroll in Engagement Plan* and then click Add. The save action is added to the Added Save Actions field.
5. In the Added Save Actions field, select the *Enroll in Engagement Plan* save action and then click Edit.



6. In the Enroll in Engagement Plan dialog box, click Select.



7. In the Select Engagement Plan dialog box, expand the engagement plan that you want to enroll the visitor in, and select the state that they should be added to.



8. Save and publish these changes to the form.

Send feedback about the documentation to docsite@sitecore.net.

Walkthrough: Creating an engagement plan

An engagement plan is a predefined plan that lets you control how your website interacts with some contacts. You use engagement plans to nurture relationships with your contacts by adapting communication to use the content, channels, and media that are appropriate for each individual interaction.

Before you create an engagement plan, you should have a scenario prepared. You should also have prepared all relevant pages, graphical elements, e-mail messages and so on.

For example, you can create an engagement plan that evaluates contacts to a website who have requested a brochure to determine whether they have also subscribed to a newsletter. If they have not subscribed to the newsletter, you can set up the engagement plan to send them a message that encourages them to sign up for the newsletter.

An engagement plan consists of:

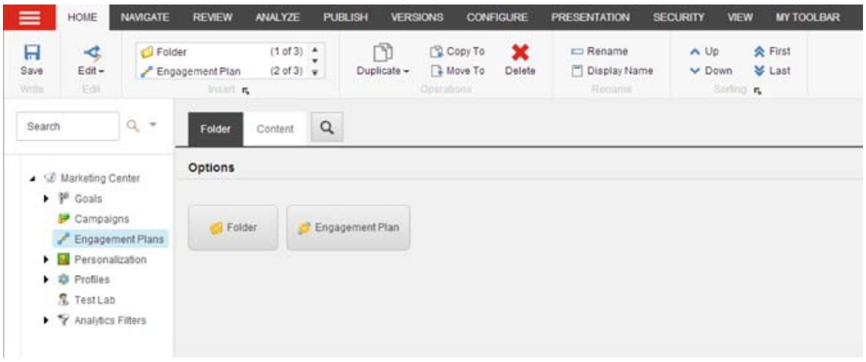
- States – the status of a website contact, such as whether or not they have visited certain pages, achieved specific goals, performed specific actions, where they have come from, and so on.
- Conditions – logic-based statements that you set which determine whether contacts in a specific state fulfill certain criteria. Conditions can relate to the contact in general or this particular visit.
- Rules – criteria that you define which determine whether a contact fulfills a condition.
- Triggers – events that determine when Sitecore should evaluate the conditions and rules associated with a state. There are two types of triggers: contact and timeout triggers.
- Actions – you set actions that should be taken when contacts meet or fail to meet the rules specified in a condition.

Creating an engagement plan

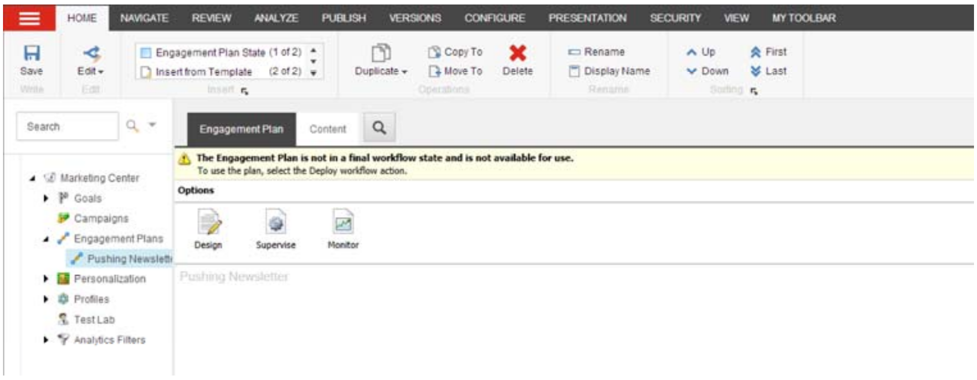
The first thing you must do is create the engagement plan item.

To create an engagement plan:

1. In the Sitecore Desktop, click Sitecore, Marketing Control Panel.
2. In the content tree, click Engagement Plans and in the right-hand pane, on the Folder tab, click Engagement Plan.



3. In the dialog box that appears, enter a name for your engagement plan, for example, Pushing Newsletter. The engagement plan appears in the right-hand pane.

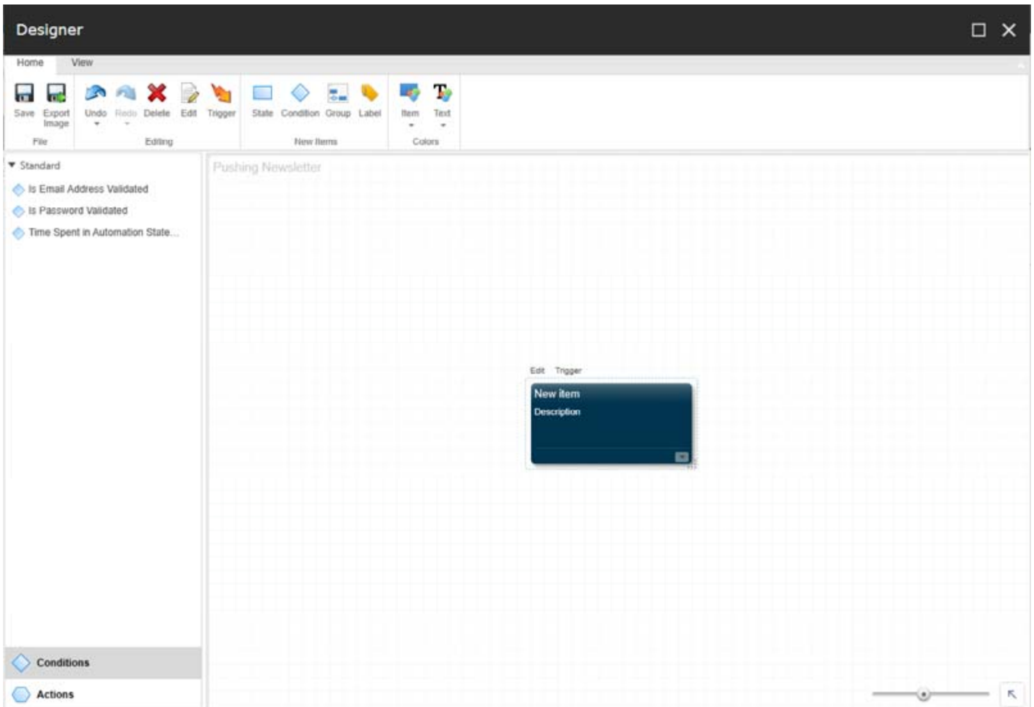


Creating the first state

After you have created the engagement plan, you need to create the first state in the engagement plan.

To create a state:

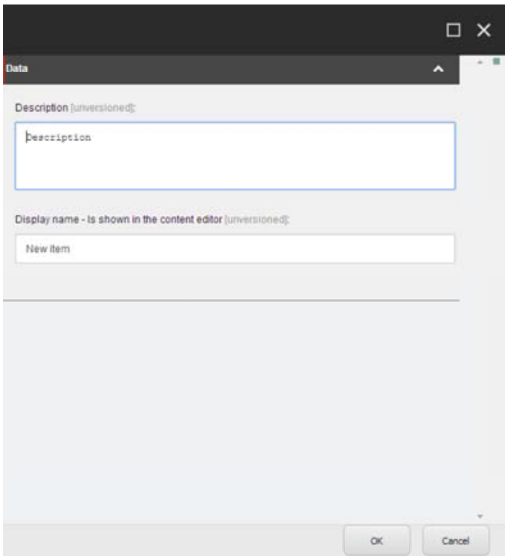
1. On the Engagement Plan tab, in the Options section, click Design.
2. In the Designer, in the New Items group, click State, and a state appears in the design pane of the engagement plan.



Note

Contacts who are enrolled in an engagement plan do not have to be placed in the first state. You can place contacts in any engagement plan state.

3. Select the new state and click Edit.



4. In the dialog box, enter a description for the state.

   For example:

Field	Value
Display name	Brochure Requested
Description	All the contacts who requested a brochure.

5. Click Save.

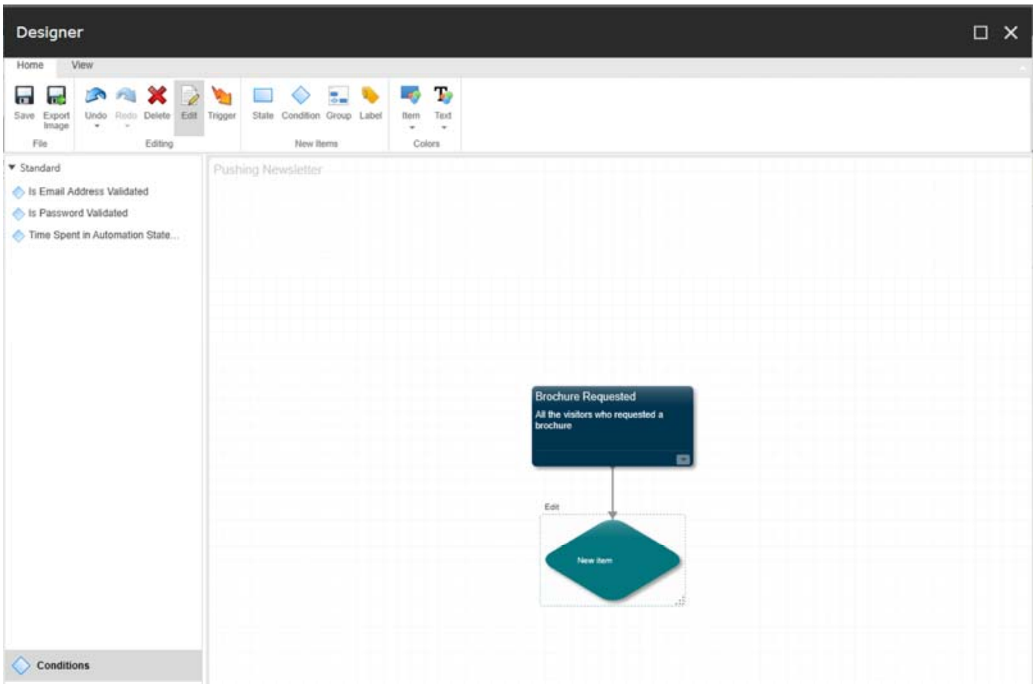
Defining a condition

After you have created the initial state, you can define the first condition that evaluates the contacts who are in this state.

To define a condition:

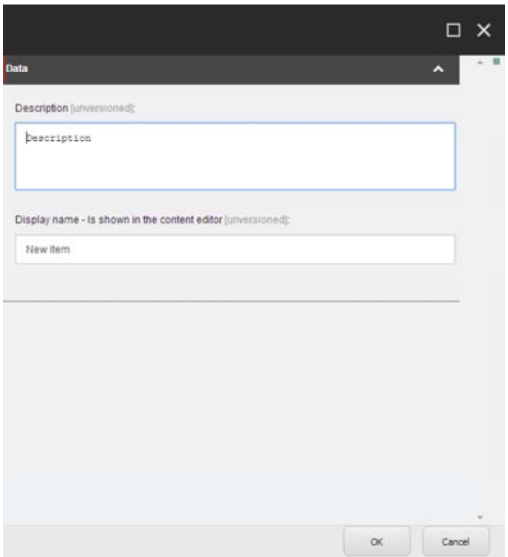
1. In the Designer, in the design pane, select the state. Shapes appear around the state.

2. Click one of the diamond shapes and a condition appears in the design pane.



This condition describe that the contacts in the initial state have requested a brochure but have not subscribed to the newsletter during this visit.

3. Select the condition and click Edit.
4. In the dialog box, enter an appropriate name and description for your condition.



For example:

Field	Value
Display name	Contact – plus brochure, minus newsletter.
Description	Has the contact requested a brochure but not subscribed to the Office Core newsletter?

5. In the Evaluate drop-down field, select a value that determines when Sitecore evaluates your condition.
6. The options are:

Option	Meaning
Always Evaluate	Sitecore always evaluates all triggers (timeout and events) for the condition.
Only Evaluate on Timeout	Sitecore evaluates the condition when one of the timeout triggers has been met: The contact has been in this state for the timeout interval that you have specified. The timeout interval that you have specified has elapsed since the last time that the condition was evaluated.

	If you select this option, Sitecore ignores the condition until the associated timeout occurs.
Only Evaluate for Subscribed Events	Sitecore evaluates the condition for a contact to the website who achieves one of the goals or activates one of the page events that are associated with this condition. If you select this option, Sitecore ignores the condition until the associated event occurs.

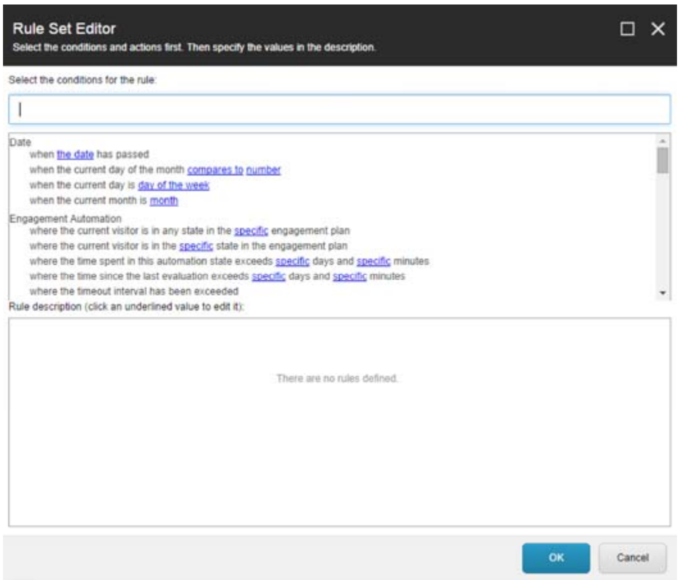
For this example, select *Always Evaluate*.

Defining a rule

Once you have defined a condition, you can define the rule or set of rules that you want triggered when the condition is met.

To define a rule:

1. In the Rule field, click Edit Rule to open the Rule Set Editor dialog box.

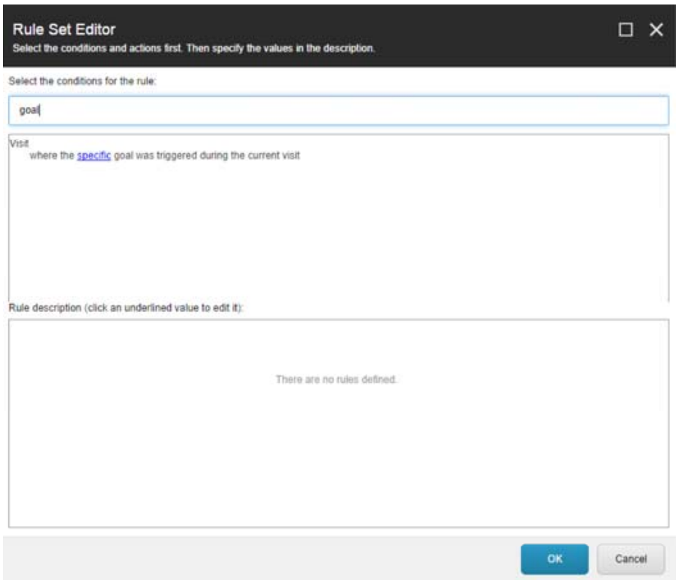


This dialog box lists all of the conditions that you can use to create a rule or a set of rules.

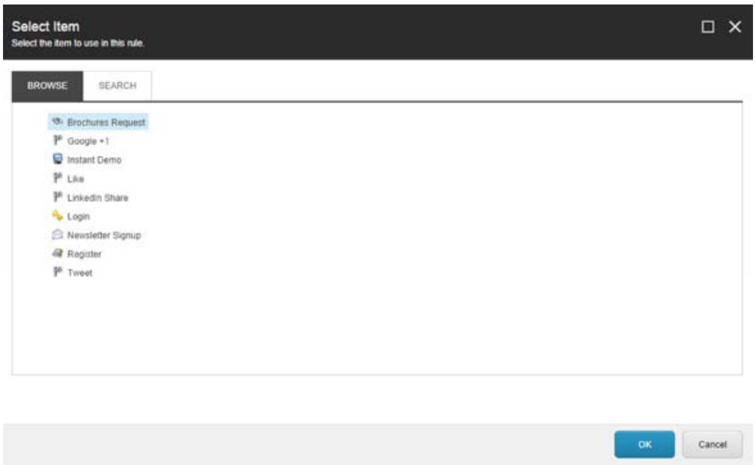
2. Select the condition for the rule that you want to implement by entering search text in the Filter field, or by scrolling through the available conditions.
3. For example, in the Select the condition for the rule field, in the Filter field, if you enter *goal*, only the rules that contain the word goal are displayed.

   Note

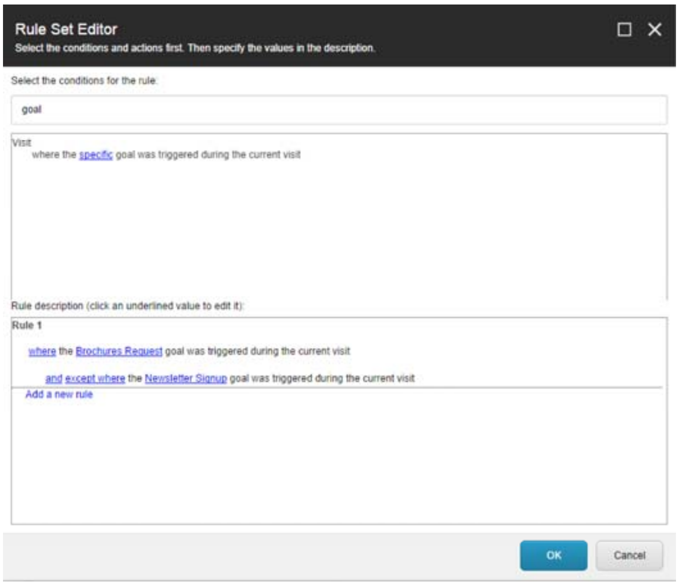
   If you want to reverse the conditions in a rule, in the Rule description field, click *where*. This automatically changes *where* to *except where*.



4. In the Rule description field, in the Rule 1 section, click *specific*.
5. In the Select Item dialog box, select the item that you want to associate with this rule, for example, *Brochure Request*.



6. To add additional conditions to your rule regarding other goals that contacts have achieved or events they have triggered, select an additional condition for the rule.
7. For example, to add an additional condition to the existing rule, in the Rule Set Editor dialog box, in the Select the condition for the rule field, select the *where the current visit has triggered the specific goal condition* again. Click on *where* to reverse the condition to *except where*.
8. If you have added an additional condition to your rule, you need to associate another item or value to the rule. Click *specific* and in the Select Item dialog box.



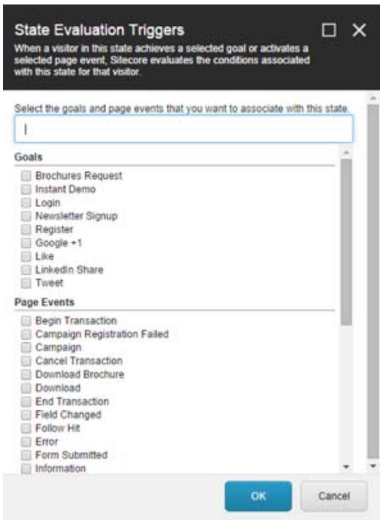
For example, if you select *except where the Newsletter Signup goal was triggered* as the value for the rule, a contact must have fulfilled the brochure request goal during the current session but not signed up for the newsletter during this session.

Defining a trigger

After you have defined the state, condition, and rules for the first state of your engagement plan, you can define a trigger to move a contact from the first to the second state.

To define a contact trigger for a state:

1. In the Designer, select the first state in the plan and then click Trigger.



2. In the State Evaluation Triggers dialog box, select the goals and page events that you want to trigger the evaluation of this state.

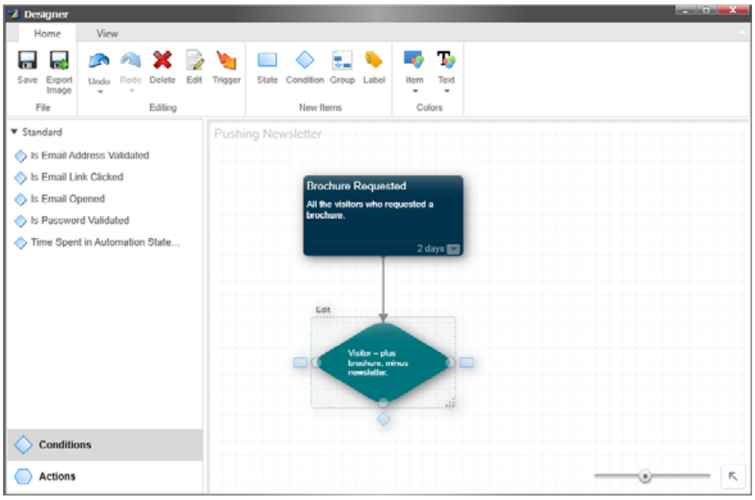
When a contact achieves one of the goals that you select, or when they activate one of the page events that you select, the engagement plan evaluates the conditions associated with this state for that particular user.

Defining the next state

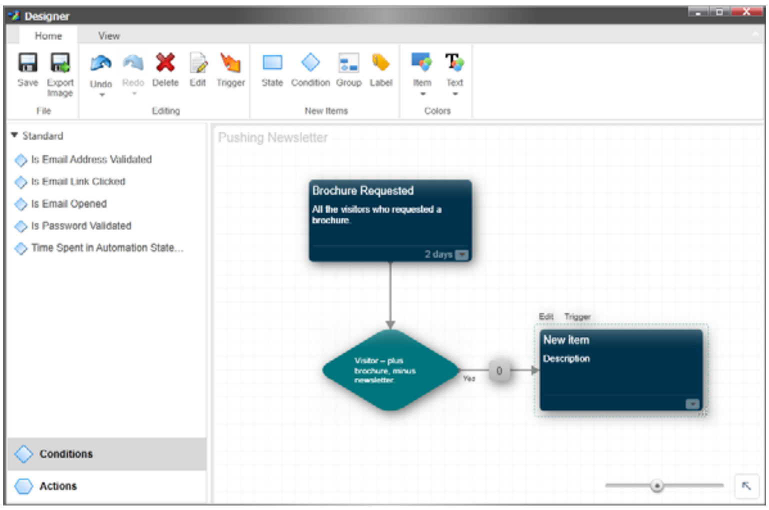
When a contact meets the conditions specified for the first state, they should be moved to the next state in your engagement plan.

To create the next state:

1. In the Designer, click the condition.



2. Click one of the rectangle shapes and a new state is added to the engagement plan.



In the Designer, you can see that there is a *Yes* beside the arrow linking the condition to the new state. This indicates that the contacts in the new state satisfy the rules that you defined in the condition.

3. Select the new state, click Edit and enter the relevant values in the dialog box

   For example:

Field	Value
Display name	Push newsletter mail recipients
Description	Contacts who have been contacted to sign up for newsletter.

4. Click Save.

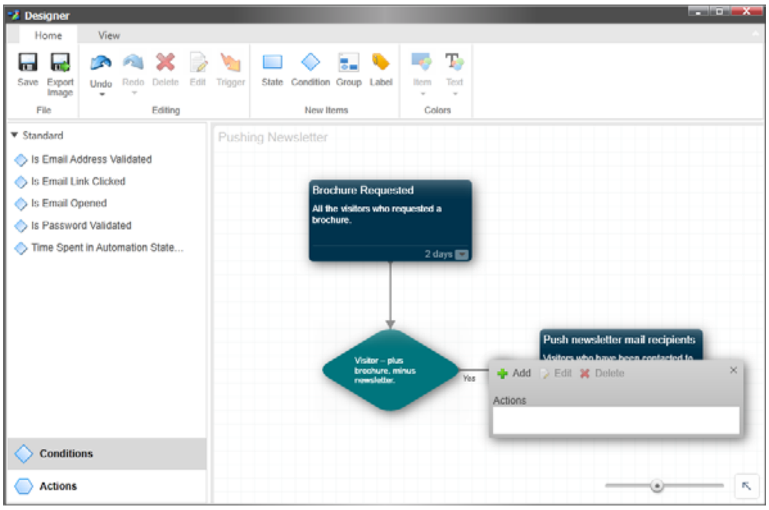
Defining an action

After you have defined the next state, you can define the action that the engagement plan should perform on the contacts who meet the conditions specified in the previous state.

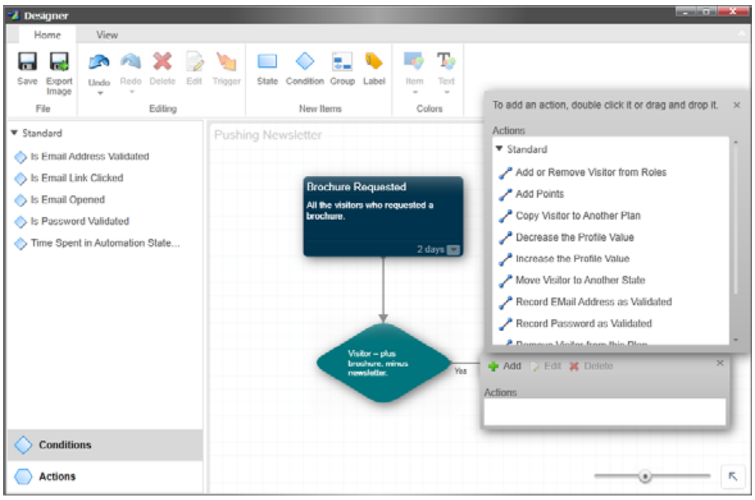
A small square in the middle of the arrow connects the condition with the new state. This is the where you define the actions that you want performed on the contacts who meet the condition that you defined for the first state.

To define an action:

1. On the arrow that connects the condition with the new state, click the square to open the Actions dialog box.



2. Click Add.

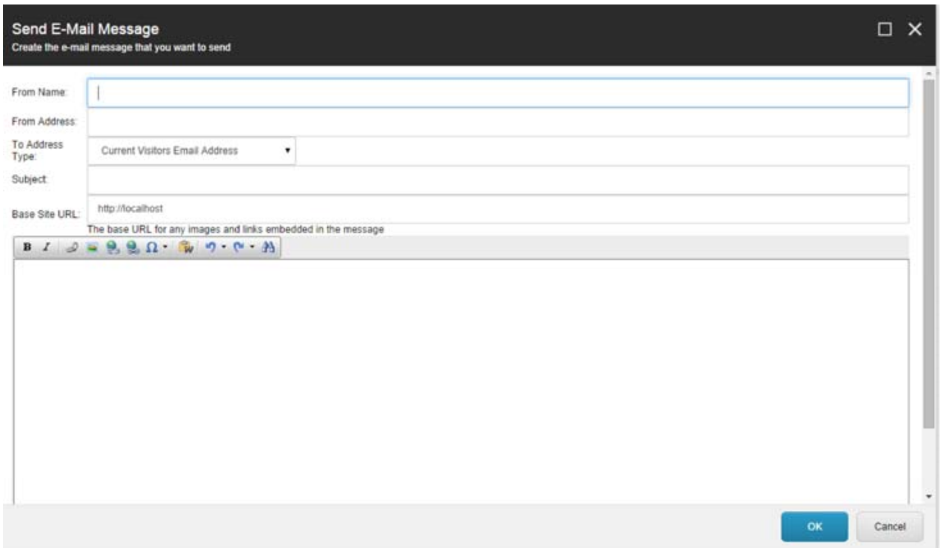


3. In the dialog box, double-click one of the available actions to add it to the Actions field.

   For example, click the *Send Email Message* action to add it to your engagement plan.

4. In the Actions dialog box, click Edit to configure the action.

   For example, if you click the Send E-mail Message dialog box, you can create a message that encourages your contacts to sign up for the newsletter.



5. In the Designer, click Save.
6. The actions box in the engagement plan now displays the number 1. This indicates that you have defined one action.

You have now configured the actions that this engagement plan performs when contacts to your website meet the conditions that you specified for the first state.

Publishing the engagement plan

You have now created a simple engagement plan. Before you can use the engagement plan on your website, you must deploy and publish it.

Engagement plans are in a workflow and you must ensure that they are in the final workflow state before you can publish them.

1. To deploy the engagement plan, in the Marketing Control Panel, select the engagement plan and on the Review tab, in the Workflow group, click Deploy.

   The engagement plan is now in the final workflow state and you can publish it.

2. To publish the engagement plan, in the Marketing Control Panel, select the on the Publish tab, click Publish, Publish item.

   The wizard will guide you through the publishing process.

An engagement plan can be very simple or much more detailed with several conditions for each state. For example, an engagement plan can contain conditions that evaluate whether contacts who received an e-mail message actually opened it, and whether they clicked the link to a newsletter subscription form that the message contains. You can use timeout triggers to send periodic e-mail messages to contacts who have not registered for the newsletter, and additionally you could use personalization rules containing criteria about the contact's location or browsing patterns to make the content contained in the e-mail more relevant.

Send feedback about the documentation to docsite@sitecore.net.

Monitoring an engagement plan

An engagement plan is a predefined plan that lets you control how your website interacts with some visitors. You use engagement plans to nurture your relationships with your visitors by adapting the communication to use the content, channels, and media that are appropriate for each individual interaction.

When you create an engagement plan, certain visitors' online activity is subject to a set of rules and conditions that determine when particular actions are performed. For example, you can create an engagement plan that evaluates contacts to a website who have requested a brochure to determine whether they have also subscribed to a newsletter. If they have not subscribed to the newsletter, you can set up the engagement plan to send them a message that encourages them to sign up for the newsletter.

In the Marketing Control Panel, you can use the Monitor to get an overview of the engagement plans that you have deployed. The Monitor is a tool that you can use to see the structure of your engagement plans. You can view how many visitors are in each state, and you can also see the percentage of visitors that are in each state of the engagement plan.

However, you cannot edit engagement plans in the Monitor, nor can you see the details of any of the states, conditions, and rules that you have defined. You can only edit engagement plans in the Supervisor.

Send feedback about the documentation to docsite@sitecore.net.

Supervising engagement plans

An engagement plan is a predefined plan that lets you control how your website interacts with some visitors. You use engagement plans to nurture relationships with your visitors by adapting communication to use the content, channels, and media that are appropriate for each individual interaction.

After you have deployed an engagement plan, you can open it in the Supervisor and see how many users are currently in each state in the plan. You can also test your engagement plans before they go live. You can add visitors to any of the states that it contains and then make the plan evaluate the conditions that you have defined and force the triggers.

In the Supervisor, you can see:

- All of the states and conditions in the engagement plan.

- The number of visitors that are in each state.
- The percentage of the total number of visitors that are in each state.

The Supervisor also displays a pie chart that illustrates the distribution of visitors in the engagement plan.

Picture 22

Send feedback about the documentation to docsite@sitecore.net.

Experience Profile

The Experience Profile lets you monitor the behavior of contacts that have interacted with your company or with your website. You can identify current as well as future customers from their very first point of contact, even if they have not provided a name.

Prerequisites

- Sitecore 8.0 or later.

  Sitecore should be preconfigured to use profiles, goals, campaigns, and automation plans before it is possible to experience the full functionality available in the Experience Profile.

The contact

In Sitecore, a contact is somebody that uses one or more devices to interact with a company or organization. The contact is a container that stores information about the behavior of customers and potential customers from the devices they use, their online interactions, such as website visits, goals converted, and campaign pages visited, and can potentially handle offline activities, such as stores visited and purchases made.

In the Sitecore Experience Platform (XP), from starting out as an anonymous visitor to a website, contacts can be used to build up a picture of potential customers. By storing data from a wide variety of sources, it is possible, over time, to build up a detailed picture of how a contact interacts with your organization.

The Experience Profile

The Experience Profile enables you and your sales teams to monitor the key areas of customer experience and interaction, such as visits, campaigns, goals, profiles, automations, outcomes, and keywords. For example, for each contact, you can see at a glance which events and goals they have triggered as well as how many engagement value points they have accumulated on your website.

The Home page of the Experience Profile is a dashboard that lets you find specific contacts. When your website has been running for some time, you may have to search through many contacts.

Picture 1

You can use the Experience Profile to focus on the accumulated experience of a single contact and view the details of all their interactions and experiences with your organization.

In the following example, you can see the contact details for the contact Andrew Baker. You can see which events he has triggered, campaigns visited, visit and engagement value statistics, as well as his best profile pattern matches. This is valuable information for marketers and sales executives.

Picture 3

Send feedback about the documentation to docsite@sitecore.net.

Experience Profile architecture and data overview

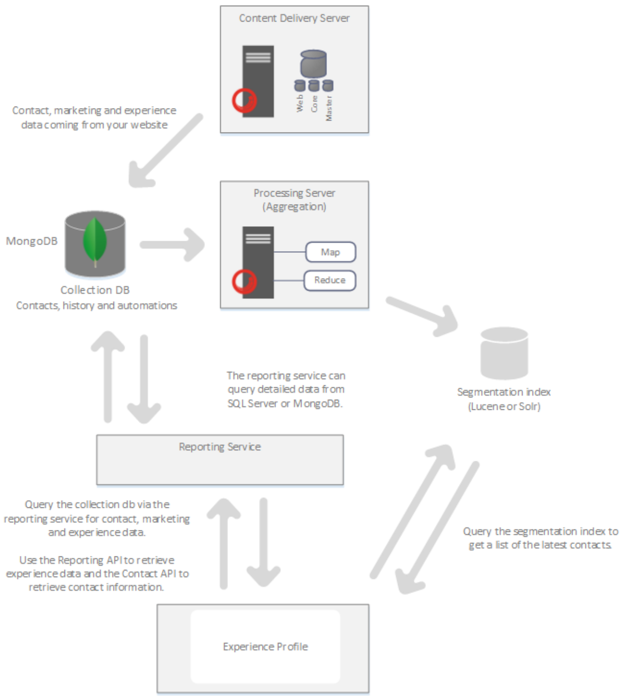
The Experience Profile enables sales and marketing employees to view experience data from the MongoDB collection database for contacts and their interactions.

Overview

You can identify current as well as future customers from their very first point of contact, even if they have not provided a name. You can examine their level of engagement and commitment with your organization by viewing the pages they have visited, profile pattern matches, goals, campaigns, and the outcomes they have achieved.

All contact and interaction data is collected and stored in the MongoDB collection database and is then aggregated by the processing layer for use by the reporting database. All marketing definition data such as goals, campaigns, page events, engagement plans, and outcomes is stored in Sitecore in the Master database. The MongoDB collection database stores the IDs of these marketing entities so it can track them, but it does not store the definition items themselves.

Experience Profile data flow



Querying contact lists

When you open the Experience Profile for the first time, a list of latest contacts is displayed in the Experience Profile dashboard. You can click on a contact or use the search box to search for a known contact if you have a name or an email address.

All experience data on contacts, and their activities is stored in the MongoDB collection database. This data is first aggregated by the processing layer and then sent to the reporting database and the segmentation index.

The Experience Profile retrieves a list of latest contacts directly from the segmentation index to display on the dashboard. It does not need to use the reporting database. The segmentation index can be implemented either as a Lucene or Solr search index.

You use a standard set of queries to build a list of latest contacts taken from the segmentation index, and when you use the search box, the segmentation index fetches contacts to display in your search results. The contacts list and contact search in the dashboard are the only parts of the Experience Profile that use the segmentation index – all other data comes from the collection database.

Querying contact data

When you click on a contact in the contacts list and open it in the Experience Profile, all the data displayed comes from the MongoDB collection database.

To populate the Experience Profile, a set of standard queries run via the Reporting Service, query the collection database and send back experience data on contacts and interactions to the Experience Profile.

You create queries as C# classes using the appropriate application programming interface which enables you to query a data source. In the xDB, there are several APIs available for querying the contacts, interactions, and outcomes collections in the MongoDB database. If you need to extend the standard functionality in the Experience Profile, you can create your own custom C# query classes and add them to the query processors defined in the `Sitecore.ExperienceProfile.Reporting.config` file.

When you have created a query and added it to the list of processors, you need to create a view to display information in the Experience Profile. A view contains a link to the query and other processors that perform actions such as transformations, sorting, and paging (see the example query for campaigns). You can extend views by adding more processors, and you can change the settings in each processor to display data in different ways.

Example query for campaigns

```
<campaigns-query>
      <processor
       type="Sitecore.Cintel.Reporting.ReportingServerDatasource.Campaigns.GetCampaigns,
       Sitecore.Cintel"/>
</campaigns-query>
```

Example view for displaying campaigns in the Experience Profile

```
<campaigns>
   <processor
      type="Sitecore.Cintel.Reporting.Contact.Campaign.Processors.ConstructCampaignsDataTable,
        Sitecore.Cintel"/>
   <processor
        type="Sitecore.Cintel.Reporting.Processors.ExecuteReportingServerDatasourceQuery,
        Sitecore.Cintel">
        <param desc="queryName">campaigns-query</param>
   </processor>
   <processor
        type="Sitecore.Cintel.Reporting.Contact.Campaign.Processors.PopulateCampaignsWithXdbData,
        Sitecore.Cintel"/>
   <processor
        type="Sitecore.Cintel.Reporting.Contact.Campaign.Processors.ApplyMasterDataToCampaigns,
        Sitecore.Cintel"/>
   <processor type="Sitecore.Cintel.Reporting.Processors.ApplySorting,Sitecore.Cintel"/>
   <processor type="Sitecore.Cintel.Reporting.Processors.ApplyPaging, Sitecore.Cintel"/>
</campaigns>
```

Reporting Service API

Use the Reporting API via the Reporting Service to retrieve experience data (interactions, campaigns, and goals) from the MongoDB collection database.

How recent is the data displayed in the Experience Profile?

The current contact session must have ended before data can be written to the MongoDB collection database. The default session timeout setting is 20 minutes. Therefore, data from the collection database populates the Experience Profile, at minimum, 20 minutes after the last session has ended.

You can change the default setting if you want data to reach the collection database more quickly. To find the session timeout setting, open the *web.config* file from your website root folder and navigate to the *<sessionState>* node and change the *timeout* setting to a lower value (in minutes).

```
<sessionState mode="InProc" cookieless="false" timeout="20"/>
```

Note

This is a system-wide ASP.NET setting used across the entire Sitecore Experience Platform, and it is not specific to the Experience Profile.

Send feedback about the documentation to docsite@sitecore.net.

Configure a page event to appear in the Experience Profile

In the xDB, a page event is something that can be triggered on your website by a visitor, or it can be triggered by the system. For example, when a visitor completes a campaign sign-up form and clicks submit, this triggers a goal conversion event and a campaign event.

In the Experience Profile, some page events are pre-configured to appear in the Latest Events or Events panels by default. You can use the Content Editor to change the page events that appear in the Experience Profile, and you can also configure the icons that are shown with page events.

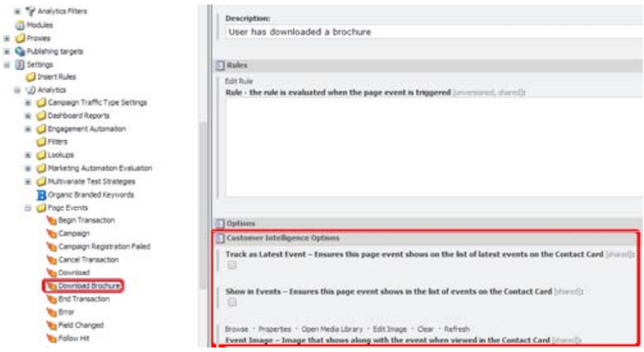
This topic outlines how to:

- Configure a page event to appear in the Experience Profile
- Add an icon to a page event

Configure a page event to appear in the Experience Profile

To configure the page events that are shown in the Experience Profile:

1. In the Sitecore Desktop, open the Content Editor.
2. In the content tree, navigate to `sitecore/system/Settings/Analytics/Page Events` and open the *Page Events* folder.
3. Select the page event item that you want to configure, for example, *Download Brochure*.



4. In the Customer Intelligence Options section, there are two check boxes for tracking page events in the Experience Profile:

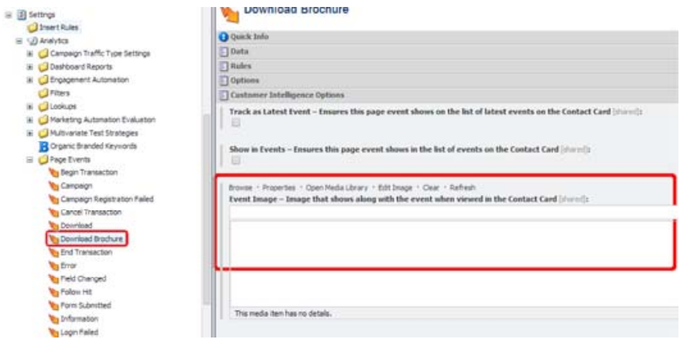
Customer Intelligence Options	Description
Track as Latest Event	When this event is triggered by the contact (visitor) it appears in the Latest Events panel on the Contact tab.
Show in Events	When this event is triggered by the contact (visitor) it appears in the list of Events shown on the Visits tab.
Event image	Associate an image with an event.

5. To make the page event item appear in the Latest Events panel, select the Track as Latest Event check box.
6. Save your changes.

Add an icon to a page event

You can configure which icons are associated with page events. To add an icon to a page event or change an icon:

1. In the Content Editor, select a page event item to configure, such as *Download Brochure*.
2. In the Customer Intelligence Options section, you have several options. You can edit the image, view its properties, open the media library, clear an image, or refresh.
3. To open the Select Media dialog box, click Browse.



4. In the Select Media dialog box, select an existing image or choose an image of your own to upload.
5. Save your changes.

Send feedback about the documentation to docsite@sitecore.net.

Configure the Experience Profile Social tab

The Experience Profile, Social tab, displays social media information related to contacts from Facebook, Twitter, G+, and LinkedIn. To see social media data in the Experience Profile, you need to have the Social Connected module installed. In Sitecore 8.0 this module is installed by default.

This topic describes how to:

- Configure the Social tab to display Twitter tweets
- Add a Twitter widget ID to Sitecore
- Verify configuration of the Twitter feed on the Social tab

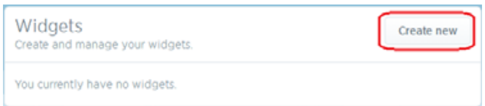
Configure the Social tab to display Twitter tweets

You need to create a Twitter widget to display tweets on the Experience Profile, Social tab. A Twitter widget is a small application that allows you to add additional features and accessibility to your Twitter account.

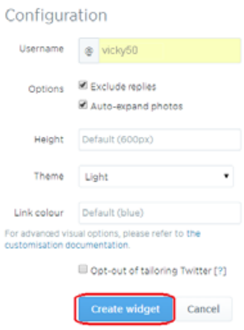
To create a widget, you can log in to Twitter with any account that you have access to. This could be your own or a company account. Once you have added a Twitter widget to the correct configuration file in Sitecore, this establishes a connection between Twitter and Sitecore.

To configure the Social tab to display Twitter tweets:

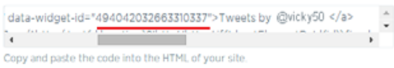
1. Log in to Twitter with your account username and password.
2. Click Profile and settings and then click Settings.
3. Click Widgets.
4. Click Create new.



5. Accept the default settings and click Create widget.



6. Copy the widget ID.



To complete the Twitter configuration, you also need to create a Twitter application.

Add a Twitter widget ID to Sitecore

To configure Sitecore to display Twitter tweets, you must add the widget ID you copied from Twitter to the social configuration file in Sitecore.

To add the Twitter widget ID to Sitecore:

1. In the Website root folder, navigate to the *Social* folder, `Website\App_Config\Include\Social`, and open the `Sitecore.Social.ExperienceProfile.config` file.
2. In the `Sitecore.Social.ExperienceProfile.config` file, find the following section:

```
<!-- SOCIAL - EXPERIENCE PROFILE - TWITTER WIDGET ID
     Specifies the widget identifier to use in the Recent Activity panel on the Social, Twitter tab.
     Default value: ""
-->
<setting name="Social.ExperienceProfile.TwitterWidgetId" value=""/>
```

3. In the value parameter, enter your Twitter widget ID. For example:

```
<setting name="Social.ExperienceProfile.TwitterWidgetId" value="494042032663310337" />
```

4. To set the number of tweets to display, enter a number in the value parameter. For example:

```
<setting name="Social.ExperienceProfile.TwitterRecentActivityTweetsCount" value="3" />
```

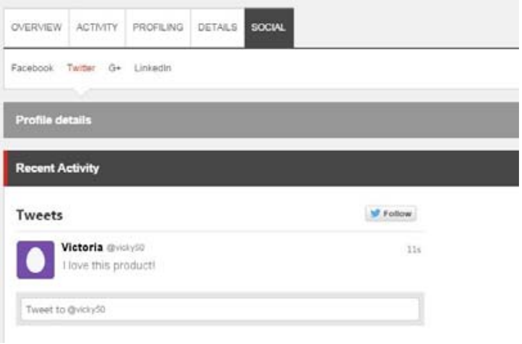
5. Save your changes.

Verify configuration of the Twitter feed on the Social tab

In the Experience Profile dashboard, you can verify that you have configured Twitter on the Social tab correctly.

To check that Twitter is configured correctly:

1. Open the Experience Profile search page.
2. Select a contact and open their Experience Profile.
3. On the Social tab, click the Twitter subtab and expand the Recent Activity panel. You should see the latest Tweets for this contact.



If you cannot see the Social tab, check that the Social Connected module is installed correctly. If you do not see any tweets check that you have followed all the steps in this topic correctly.

Send feedback about the documentation to docsite@sitecore.net.

Experience Profile configuration

The Experience Profile is one of the core Sitecore applications that is available in the Launch Pad, so there is nothing to install. If you have the appropriate permissions, the Experience Profile works straight out of the box. However, if you have not configured any content profiles, goals, or campaigns in Sitecore, the experience data that you see will not be very meaningful.

Configuration

Before you can start using the Experience Profile:

- Create and set up a new role that uses the *Analytics Reporting* role (security administrator task).
- Configure Sitecore digital marketing functionality. For example, if you have not implemented a profiling strategy, set goals, created campaigns and engagement plans, or had any site visits then there will only be a limited amount of information available about your contacts.

If you do not configure any experience marketing functionality, you will still be able to see a list of contacts (visitors) but you will only see general information about these contacts, such as the external keywords used and general session information. Without some key metrics, such as goal conversions or engagement value accumulated, it is difficult for sales staff to determine the levels of commitment exhibited by the contact and whether they are likely to become a future customer.

To configure Sitecore XP to work with the Experience Profile, you need to follow a series of steps that require careful planning. You can find documentation on doc.sitecore.net that can help you to learn how to configure to create profiles, implement predictive personalization, set goals, or set up MV tests in line with the guidelines set out in the Customer Experience Maturity Model.

To preconfigure Sitecore for the Experience Profile:

- Evaluate your website
- Create personas in the Sitecore Marketing Center
- Create and configure profile cards
- Allocate profile values to the content on your website
- Create and configure a set of pattern cards
- Create goals
- Create campaigns
- Configure personalization
- Create automation plans
- Collect visits on your website

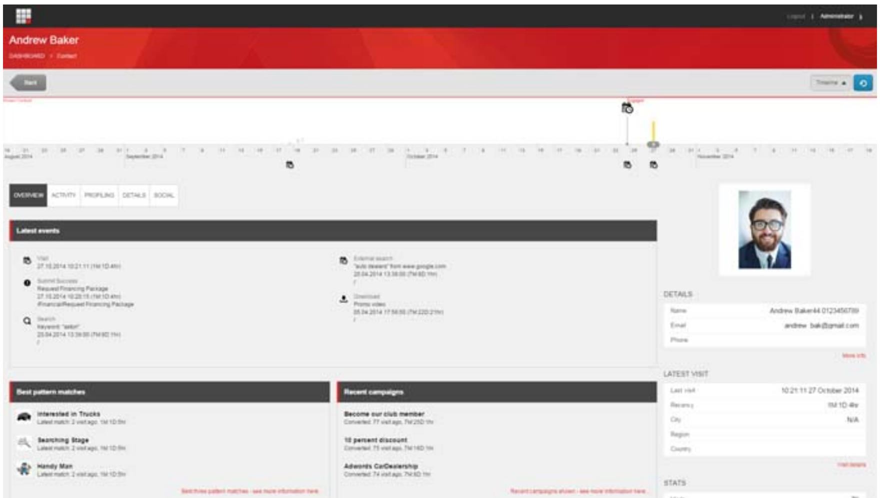
Send feedback about the documentation to docsite@sitecore.net.

Experience Profile dashboard

The Experience Profile consists of a timeline and multiple tabs that display a summary of key experience analytics data for individual contacts. You can view one contact at a time.

An information area to the right continually displays a summary of the latest visit information and metrics associated with the contact.

The *Latest Visit* panel contains a link to the *Visit Detail* report, which displays detailed information about the contact's latest visit.

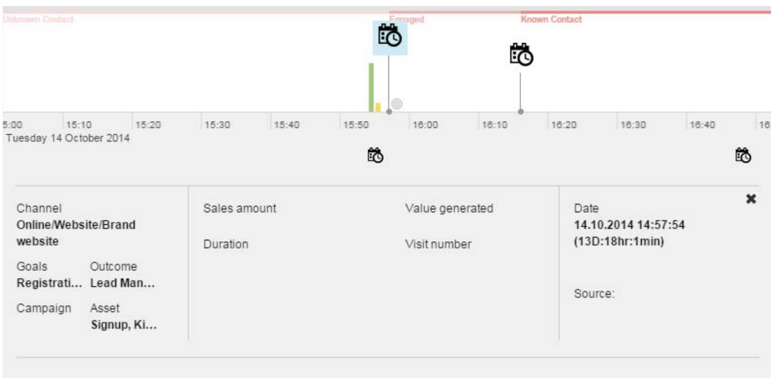


Experience Profile timeline

Across the top of the Experience Profile, a timeline displays the Experience Marketing events and activities that a contact has had with an organization. A series of icons highlight important events on the timeline. Click an icon to see details of the event displayed below the timeline. For example, which channel brought the contact to the website.

Icon	Description
	New contact – shows the date and time of the first interaction, even if the contact was anonymous at this stage.
	Channel – indicates from which channel a contact originated. This icon is used by default if an icon has not been assigned to the channel.
	Goal conversions – this icon is displayed when a goal is triggered.
	Duration of visit – date, time, and length of the contact session.
	Value – engagement value points accumulated by the contact at this point in the timeline.
	Campaign – this icon is displayed when a campaign event is triggered.

To drill down into the timeline, click on any of the activities displayed. For example, if you click the clock icon you can see more information about the channel that the contact used to make the interaction.



Experience Profile tabs

The following table contains an overview of the information contained on each tab and subtab in the Experience Profile:

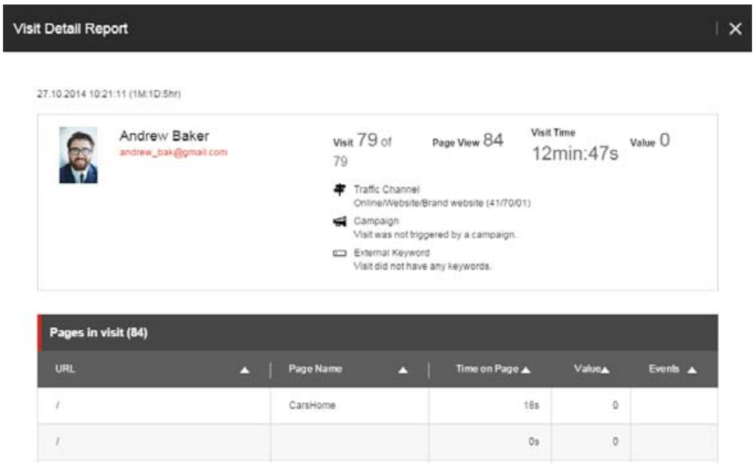
Tab/Subtab/Group	Description
Overview/Latest events	

	Displays the latest contact visit and up to 9 other latest events. For example, goals converted, downloads, or external/paid searches. You preconfigure the events that appear in the Latest events group in the Content Editor.
Overview /Best pattern matches	Displays the top 3 best pattern matches for a contact based on how closely a pattern matches the visitor's profile.
Overview/Recent campaigns	The 3 most recently triggered campaigns.
Activity/Visits/Visits	Displays a list of all the visits sorted by recency. Click a visit to open the *Visit Detail Report*. The *Visit Detail Report* contains more detailed information about the visit, such as the pages visited, the goals converted, and the internal search terms used.
Activity/Visits/Events	Displays a list of the latest events that the contact has triggered. The events are sorted by recency. Events that appear in the Events group are preconfigured in the Content Editor.
Activity/Outcomes	Lists all the outcomes the contact has achieved. Shows the stage in the default Lead Management Funnel that the contact has achieved including the date and time that different outcomes were achieved.
Activity/Campaigns	Lists all the campaigns that the contact has participated in.
Activity/Channels/Distribution	Displays a graphical representation of the activities that the contact has engaged in and their distribution across the different channels. The channels are: • Direct • Digital Events • Tele Marketing • Store
Activity/Channels/Summary	A list of the activities by channel.
Activity/Goals	Lists all the goals that the contact has triggered. •
Activity/Automations/Engagement plans	Lists the engagement plans that the contact has triggered and the current state the contact occupies in each plan.
Activity/Keywords/External keywords	A list of the distinct keywords that the contact used in search engines, such as Google or Bing. For selected keywords, you can see more detailed information, such as a list of the search engines, visit duration, and value.
Activity/Keywords/Internal keywords	Lists the keywords used in searches on the website.
Activity/Keywords/Paid keywords	Lists the distinct paid keywords used, for example, from Google AdWords. For selected keywords, you can see more detailed information, such as a list of sources, visit duration, and value.
Profiling	Displays the best pattern card matches for each profile based on visits made by this contact. This tab displays the best pattern card matches for each profile based on the visits this contact has made to the website. For example, you could have the following set of profiles: • Persona • Car Type • Stage in the buying cycle For each profile, a radar chart and profile key scores are displayed. Profiles and profile keys are preconfigured in the Sitecore Marketing Control Panel.
Details/Contact details	Contains all the contact information for the contact. Contacts enter all the email addresses and phone numbers themselves.
Social/Facebook	If you have configured a connection to Facebook, this subsection displays a summary of the contact's Facebook information.

Social/Twitter	If you have configured a connection to Twitter, this subsection displays a summary of the contact's Twitter information.
Social/G+	If you have configured a connection to G+, this subsection displays a summary of the contact's G+ information.
Social/LinkedIn	If you have configured a connection to LinkedIn, this subsection displays a summary of the contact's LinkedIn information.

The Visit Detail Report

The Visit Detail Report displays more details about the contact's website visit or session. For example, a list of pages viewed during visits with URLs, external keywords used, and details such as engagement value accumulated.



Send feedback about the documentation to docsite@sitecore.net.

Experience Profile FAQs

Commonly asked questions on using or configuring the Experience Profile, which enables you to monitor the key areas of customer experience and interaction, such as visits, campaigns, goals, profiles, automations, outcomes, and keywords.

Which events are displayed by default on the Overview tab?

Any event that a system administrator has configured to be triggered appears in the Latest events panel. To see which events have been configured to appear, in the Content Editor, navigate to: */sitecore/system/Settings/Analytics/Page Events*. Select an event and view the Customer Intelligence Options section. Before you use this application for the first time, you need to understand what is required for Experience Profile configuration.

How many Latest events are displayed by default?

By default, a maximum of 10 Latest events can be displayed on the Overview tab.

What does the Latest events group consist of?

The Latest events group consists of the top 10 latest events, including the last contact visit plus another 9 distinct types of events. These can include any page event, external keywords, and paid search terms used. The default latest events are preconfigured in the Content Editor to appear on the Overview tab in the Latest events group.

How many Best Pattern Matches or Recent Campaigns are displayed by default?

By default, a maximum of 3 pattern matches and recent campaigns are displayed on the Overview tab.

Which image file types can be used as icons?

You can use all the usually allowed image files types, such as JPEG, PNG, or GIF.

Is there a recommended image size or resolution for icons?

There is no recommended image size but try to keep icons small with a low resolution.

What is the difference between an event and a latest event?

The only difference between events is the way in which they are configured in the Content Editor to be displayed in the Experience Profile dashboard.

In the Content Editor, you can configure events to appear in two different ways:

- Track as Latest Event – The event is displayed in the Latest events group on the Overview tab.
- Show in Events – The event is displayed in the list of events on the Activity - Visits tab.

Is it possible to see the session trail for a single contact?

Yes, view the *Visit Detail* report to see details of the session trail. This includes number of visits, pages viewed, search terms used, goals converted, and engagement value generated.

Why can't I see any campaigns in the Experience Profile?

Only campaigns triggered on a landing page (the first page visited) appear in Recent campaigns. It is possible that some campaigns for a contact were triggered during the visit and not on the landing page. If you want a campaign triggered during a visit to appear in the Experience Profile, you can use the Content Editor to configure campaigns events. In the Content Editor, select the campaign event you want to show and in the Customer Intelligence Options section, select the Show in Events check box or Track as Latest Event check box.

Can I see all the devices used by a contact?

No, the Experience Profile does not display any specific device information for a contact yet.

Can I search for a contact using phone number, address, or organization?

No, you can only search for a contact by name or by email address.

Why in the Visit Detail report can I see that the accumulated engagement value is 115, but the sum of all goals in this visit is only 55?

It is not just goals that generate engagement value points. As well as triggering goals, it is also possible for a contact to trigger other events that have engagement value points assigned to them during a visit.

Is it possible to see the referred site for visits with the channel "Referred-…"?

No, it is not possible to see the referred site for this channel.

Send feedback about the documentation to docsite@sitecore.net.

Experience Profile sales executive scenario

The Experience Profile dashboard provides an overview of the levels of engagement and commitment shown by a contact, which in turn can help sales executives to understand the characteristics, browsing behavior, and buying preferences of a contact.

In this example scenario, you want to get to know a named contact and find out how engaged he is with your car dealership website. To see the session trail for all interactions and activity, you can view triggered goals and events, campaigns that have been enrolled for, triggered engagement plans, and keywords used.

In this scenario, you complete the following tasks:

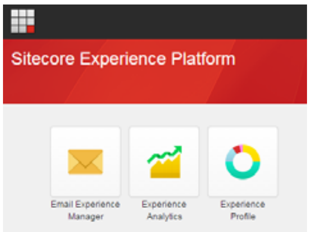
- Open the Experience Profile dashboard and find a named contact
- Use the Experience Profile to get to know a contact
- Analyze the current activity level of a contact

Open the Experience Profile dashboard and find a named contact

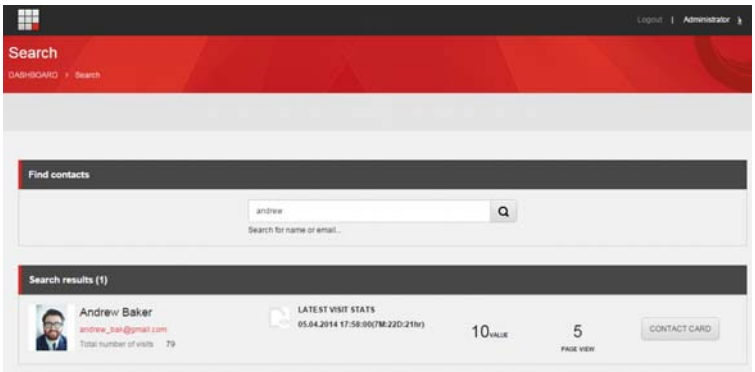
You can find a contact by entering a name or an email address in the search box provided in the Experience Profile.

To open the Experience Profile and find a named contact:

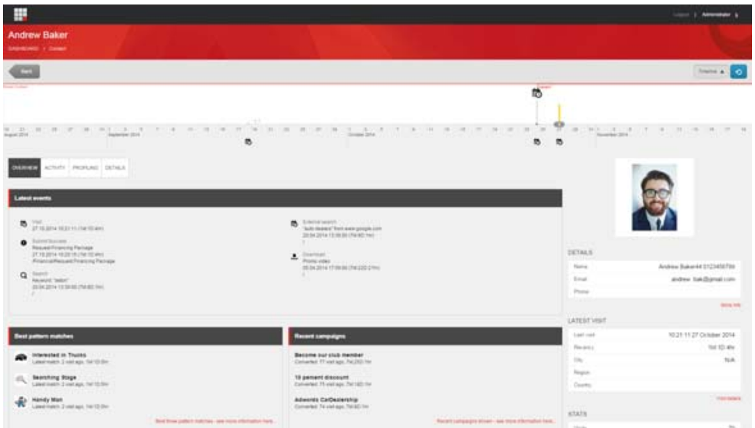
1. In a web browser, enter the URL of your Sitecore website.
2. Enter your log in credentials and click Log in.
3. To find contacts, click Experience Profile. By default, a list of the latest visitors to your website is displayed.



4. You can select a contact directly from the list or alternatively use the search box to find a known contact by name or email address.
5. For example, enter the name *Andrew Baker*, his email address, or part of the name in the Find contacts search box to find out more about the experience this contact has had with your website. Alternatively, click his name if you can already see it displayed in the list of contacts.
6. Look for the relevant contact in the search results and then click Contact Card to display the xFile for this contact.



7. In this example, the Experience Profile for *Andrew Baker* is displayed. By default, the Overview tab is displayed first.



Use the Experience Profile to get to know a contact

The Experience Profile dashboard provides a quick summary of the key interaction data for each contact that has visited your website. This data could include visits, pages viewed, enrolled campaigns, engagement value points generated, engagement plans, and social media information.

The Experience Profile can help you gauge the level of interest a contact has shown in your website. You can target promising sales leads and follow up initial interest with a telephone call or meeting. It is important to know whether a contact is merely browsing pages on a website without any intention of buying or if they are showing more signs of commitment, for example, by completing a campaign registration form or by downloading a price list or brochure. The Experience Profile provides this valuable information. It also might be possible to determine from the Experience Profile whether the contact is a premium customer interested in top-of-the-range products.

The following selection of tabs and subtabs in the Experience Profile dashboard demonstrates how you can analyze the level of engagement of a specific contact.

Overview tab

On the Overview tab, you can get a quick overview of the activity and experience of a contact or visitor to your website:

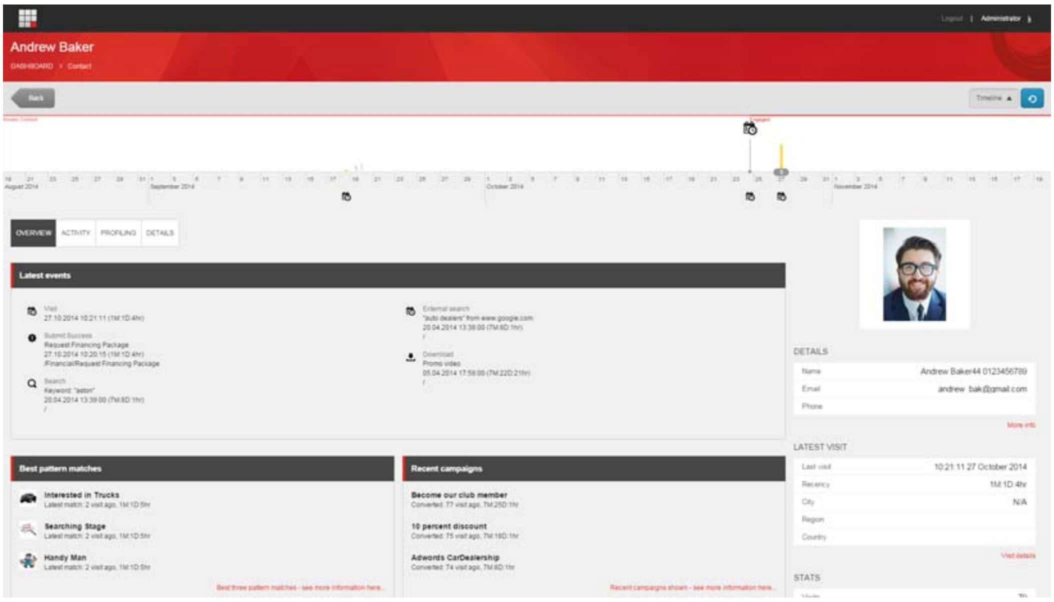
- Latest events – displays a selection of events that the contact has recently triggered on your website.

  For example, for the contact Andrew Baker, you can see his last recorded visit, that he has downloaded a promotional video, and that he has used the search keywords *aston* and *auto dealers*.

- Best pattern matches – presents an instant snapshot of the preferences and interests of the contact based on profile values and pattern matches collected during a visit. Each pattern card contains a predefined set of profile values so that as a contact browses the website they can be matched to different pattern cards, depending on the content they have viewed.

  In this example, you can see which type of car the contact is interested in and how close they are to making a purchase. The see more information link enables you to go directly to the Profiling tab to view more detailed information.

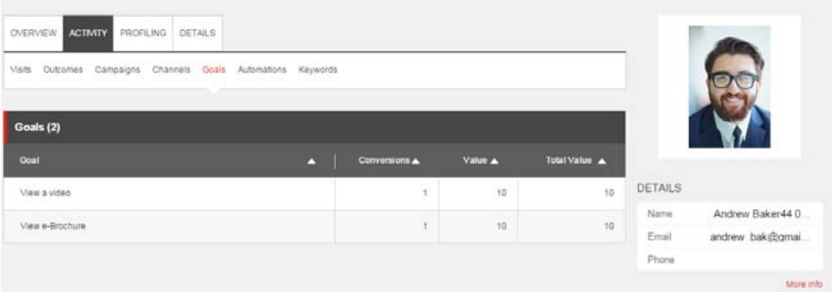
- Recent campaigns – displays the most recent campaigns that the contact has signed up for. To go directly to the Campaigns tab and see at a glance the level of commitment and engagement a contact has with your brand, click see more information here.



To the right of the Overview tab, you can see the name and email address of the contact. To see more detailed information about the contact on the Details tab, such as the location of the contact's home address, click More info.

Goals subtab

On the Activity tab, on the Goals subtab, you can view a list of goals converted along with other metrics, such as total conversions, recency, and engagement value collected across all visits. You can use this information to learn a lot about the intentions of the contact, based on which goals have been converted, particularly if the accumulated engagement value is high across a small number of visits.



In this example, you can see that some of the goals that Andrew Baker has converted include viewing an e-brochure and viewing a video. He has accumulated 20 engagement value points (total value) during his visits to the car dealership website.

Andrew Baker has downloaded several resources over a small number of visits, which indicates he is interested in buying a car. He could be put into another state of an engagement plan as a result of triggering these goals or, alternatively, a sales representative might offer to call or email him to help him to make a decision.

Outcomes subtab

On the Activity tab, on the Outcomes subtab, you can see a list of any interactions that have had a significant monetary or non-monetary outcome.

These can indicate whether you should take any further action, such as creating personalization or adding a contact to or changing an engagement plan.

The *Lead Management Funnel* is the default outcomes funnel and contains several stages.

In this example, you can see that Andrew Baker has achieved several different outcome definition levels, moving from *Unknown contact* in April to *Contract signed* in May to *Engaged* in October.

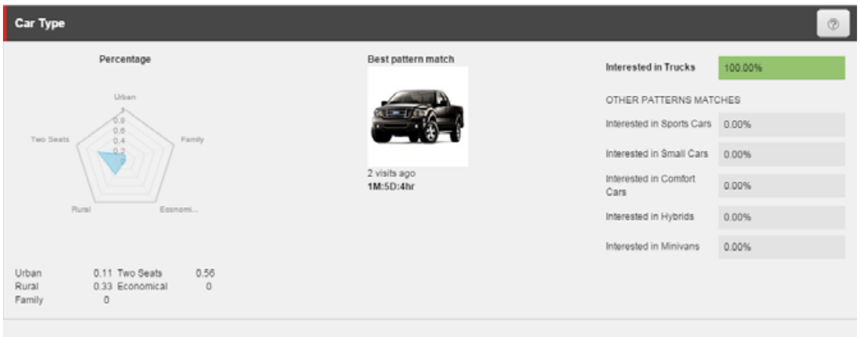
Outcomes

Icon	Date	Category	Outcome	Monetary va...
	24.10.2014 15:04:10	Lead Management Funnel	Engaged	
	20.05.2014 13:44:00	Contract	Contract Signed	20000
	05.04.2014 18:03:00	Lead Management Funnel	Known Contact	
	02.04.2014 14:44:00	Lead Management Funnel	Unknown Con...	

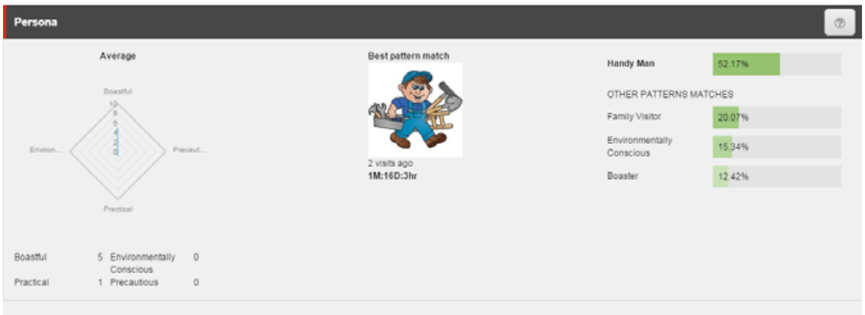
Profiling tab

On the Profiling tab, you can see a set of profiles that reflect the current interest the contact has in the content on the website. This is dynamic profile information that changes and is adapted with each page the visitor views or each action the visitor takes on the website.

In this example, Andrew Baker has been identified as being interested in trucks.



He has also been identified as matching the *Handy man* persona.



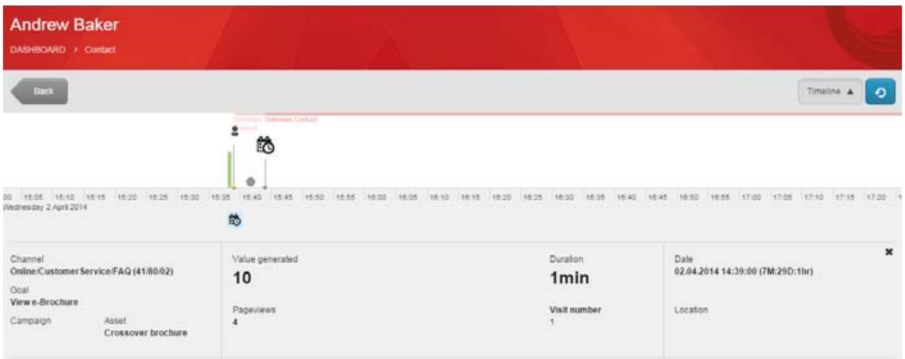
On the Overview tab, you can see that he used the search term *aston,* which suggests he has an interest in luxury cars and is in the searching stage of the buying cycle.

Profile information changes and adapts each time the contact makes a visit, so that pattern matches are dynamic and therefore constantly change. They reflect the content that the contact has viewed at that moment in time.

Analyze the current activity level of a named contact

In the Experience Profile, the timeline enables you to view a contact's activity with your brand in chronological order. You can see the first point of interaction as an anonymous contact and you can see when a visitor provided a name or an email address and became a named contact. At a glance, the timeline gives you a visual overview of all the activity that a contact has had with your brand. You can use the timeline to assess whether a contact's activity on your website is declining and evaluate whether it is time to get in touch to remind the contact of your brand.

For example, on the timeline for Andrew Baker, you can see when he first became a contact and his recent activity. His first activity as an unknown contact was viewing an e-brochure that was part of a campaign, which generated 10 engagement value points.



In this example, the timeline shows that Andrew Baker has engaged with the car dealership via a series of interactions over several months and has gone from being an unknown contact to being an identified contact and to being engaged.

His last two interactions show that he has requested finance, indicating that he is coming close to buying a product from you.

However, since the 27th October he has not shown any activity at all.

Conclusion

In this scenario, Andrew Baker has visited the car dealership website several times, downloaded content, converted multiple goals, and signed up for several campaigns. Each time his visits have shown an increasing level of commitment. His profile information shows that he is interested in *trucks*, matches the *Handy man* persona and is likely to be a premium customer. His activity on the website remained high for several months but has declined recently.

The Experience Profile can help sales or marketing executives to decide what action to take next. In this example, the contact, Andrew Baker, may respond well to a personal telephone call from a sales consultant to discuss in more detail which model of car would suit him best. A sales or marketing employee might also decide to improve the Experience Marketing strategy by asking a developer to amend or create a new engagement plan that is more suitable.

Send feedback about the documentation to docsite@sitecore.net.

Federated Experience Manager - a functional overview

The Federated Experience Manager (FXM) is a dedicated SPEAK application that lets you track and personalize the experience of a website visitor across multiple (federated) websites, including non-Sitecore websites.

FXM lets you:

- Track every visit to an external website and personalize the content on the external website by deploying a single JavaScript tag.

  The information from the external website is available in the Experience Analytics dashboard.

- Easily create the FXM JavaScript tag that you deploy on the external website.
- Enrich and personalize every visit with Sitecore marketing attributes, such as goals, events, profiles, and campaigns.
- Capture all the visitor information from the external website and store it in the Sitecore Experience Database (xDB).
- Integrate the external website with the marketing capabilities that are available in Sitecore, such as goals, events, profiles, campaigns, engagement automation, and so on.
- Use the Sitecore Experience Editor to edit the pages and their specific elements on the external website that you want to apply Sitecore marketing attributes to.
- Use and extend a dedicated API to meet your specific tracking and personalization requirements.

FXM features

- A single JavaScript tag that can be deployed on the external website.

  The JavaScript tag:

  - Acts as a beacon that is registered by the Beacon handler on the Sitecore platform. The Beacon handler manages all the requests from the external website.

    You must deploy the JavaScript tag on every page event of the external website that you want to monitor.

  - Can be deployed on every page of the external website without degrading performance.
- Tracking works with first-party domain cookies.

  Third-party cookies will be supported in a later release.

- The dedicated FXM Web API captures external visitor information and writes the information back to the xDB.
  - The FXM Web API can scale to millions of external site pages.
  - Developers can extend the FXM Web API to meet their specific requirements.
- The JavaScript API integrates with the standard Sitecore marketing functions.
- In Sitecore 8.0, FXM uses the Sitecore rules engine to define the external websites that you want.
  - The external site domain is matched through a domain matcher rule, so that only visits to that specific website are tracked in the xDB.

    For example, to track an external website to the domain www.external-demo.com, you should setup a domain matcher rule in FXM that only matches visits to www.external-demo.com.

  - All subsequent FXM features are derived from the domain matcher rule.

- FXM supports multiple languages through specific rules that you define to work with the externally tracked domain.
- All the FXM features are available in the Content Editor.
  - The Marketing Control Panel contains a node called Federated Experience Manager.
  - You can define rules through the standard item hierarchy.

FXM supports matching against any of the following conditions:

- Page or pages through standard regular expressions
- External actions, such as:
  - Capture Click – when a visitor clicks any element during their visit to an external website
  - Fragments – when a visitor interacts with a DOM fragment
  - Submit Matcher – when a visitor submits a form on an external website, such as a sign-up form
  - Placeholder Replacer – replacing content on an external website through a standard Sitecore placeholder

Managing FXM

The FXM application is available on the Sitecore Launchpad. It is easy to set up the tracking of an external website.

- The application automatically creates a domain matcher rule based on the external website domain.
- The application provides you with the JavaScript tag that the webmaster of the external website can deploy to enable FXM to track the website.

  FXM tracks every page that contains the JavaScript tag.

- FXM automatically integrates with Sitecore Experience Analytics.

Administrator features

FXM lets administrators:

- View a list of all the external sites that are tracked.

  The list of external websites displays the status of each domain, including:

  - Name
  - Created by
  - Website domain
  - Status – Published or Not Published
  - Date modified
  - Overall number of visits to the external site
- Edit external domain information.
- Open the external domain in the Experience Editor in FXM.
- Manage the individual sites from within the FXM application.
- View and manage any FXM functions that are associated with the external website.

FXM Experience Editor

The FXM Experience Editor lets developers:

- Use a dedicated web browser proxy to browse the external site.
- Add a new page filter to track specific pages - selected by browsing the site.
- Add capture click actions to trigger an action when the user clicks any HTML element on the external site.
- Replace elements on the external site with Sitecore placeholders.

The FXM Experience Editor lets marketers:

- View any external webpage elements that have been assigned FXM functions.
- Assign any Sitecore marketing attributes to these functions.
- Create all the FXM functions they need as items in Sitecore.
- Use standard Sitecore marketing functionality to assign marketing attributes to the definition items for capture click actions on the external site.
- Use standard Sitecore personalization controls to personalize placeholders.

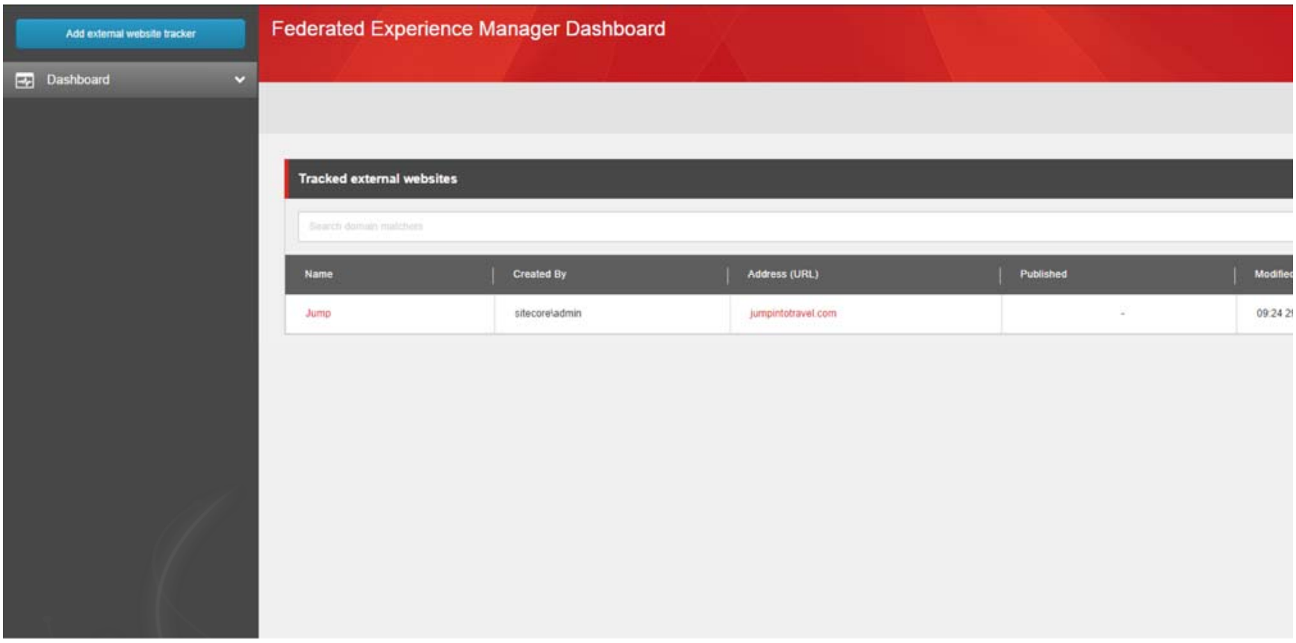
Send feedback about the documentation to docsite@sitecore.net.

The Federated Experience Manager

The Federated Experience Manager (FXM) is an application that allows you to track visitor interactions and generate analytics information on external, non-Sitecore websites with Sitecore. You can use FXM to implement personalization rules, create goals and events, and implement content profiling on an external website. You can also implement rules that include non-Sitecore sites in engagement plans, as well as track traffic from the external website in Experience Analytics.

To enable FXM functionality, a Javascript tag is added to the external websites. This tags the external site and allows the FXM application to connect the external site's components to the Sitecore Experience Platform. FXM has a dedicated Experience Editor that allows you to assign marketing attributes, implement personalization rules and personalized content as you would on the Sitecore XP.

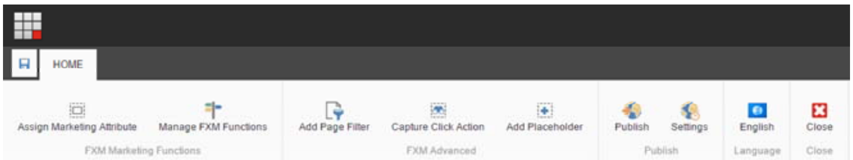
You can access the FXM through the Marketing Control Panel or through the FXM application on the Sitecore Launchpad.



Adding Sitecore functionality to an external site with the FXM

Once you have applied the FXM Javascript tag to an external website, you can assign FXM functionality in one of the following ways:

- Capture Click Action – enables you to assign Sitecore marketing functionality to external website components on a page level. For example, you use click actions in the FXM Experience Editor to add the ability to assign goals or events to a button on a landing page.
- Add Placeholder – replaces existing external components with a Sitecore placeholder, enabling you to add personalized content on the external site. For example, you can replace a banner on an external website with a Sitecore banner. You can then create different variations of the banner to show to visitors who meet certain personalization rules.
- Add Page Filter – lets you assign goals and edit components on individual pages. You can assign marketing attributes directly to a page filter. For example, you can assign a page filter, then assign marketing attributes, such as profile cards, to the filter. You can then add a placeholder on the page, and implement personalization rules that show different content to visitors who match that profile card.
- Assign Marketing Attributes – assigns marketing attributes, such as goals or events, to items that have click actions and placeholders associated with them.



Send feedback about the documentation to docsite@sitecore.net.

Configure shared session state for FXM to use SessionIDManager

In the Federated Experience Manager (FXM), you can track contact sessions and activity on non-Sitecore websites. This enables you to capture, record, and process all of a contact's activities and experiences across all your websites.

For the FXM session state to work correctly you must configure it to use its own `SessionIDManager` for FXM tracking requests on content delivery and content management servers.

Note

These steps are only applicable to Sitecore 8.0.

The FXM needs to use its own `SessionIDManager`:

- To track a contact session on an external website, the contact ID must be saved with the interaction in xDB and included in any tracking requests to the FXM service.
- Security restrictions in different browsers and in different domain scenarios mean that a contact's ID cannot always be transmitted using cookies.

To implement this change, regardless of the session state provider you use (Redis, SQL Server or MongoDB), you need to add a `sessionIDManagerType` attribute to the `sessionState` section of the `web.config` file. This ensures that FXM uses its own `SessionIDManager` for FXM tracking requests. For any other request, the pipeline will choose the default ASP.Net session ID manager.

Note

If FXM is disabled, this change will have no effect on your session state settings providing the FXM dll files are present in the website bin folder.

To configure shared session state to use the `SessionIDManager` for FXM:

1. Open the web.config file from the `wwwroot` folder of your website.
2. In the `web.config` file, navigate to the session state node:

```
<sessionState
```

```
mode="InProc"
cookieless="false"
timeout="20"
sessionIDManagerType="Sitecore.FXM.SessionManagement.ConditionalSessionIdManager">
  <providers> … </providers>
</sessionState>
```

3. Add the following line to the `sessionState` node:

```
sessionIDManagerType="Sitecore.FXM.SessionManagement.ConditionalSessionIdManager
```

4. Save your changes.

Send feedback about the documentation to docsite@sitecore.net.

Configuring FXM

The following configuration files for the Federated Experience Manager (FXM) are stored in the `/App_Config/Include/FXM` directory:

- `Sitecore.FXM.Bundle.config` – the configuration settings for the beacon script and for bundling – minification and caching
- `Sitecore.FXM.config` – the main configuration settings for FXM
- `Sitecore.FXM.Lucene.DomainsSearchIndex.config` – the configuration settings for the FXM Lucene search index that is used by the SPEAK application
- `Sitecore.FXM.Solr.DomainsSearchIndex.config` – the configuration settings for the FXM Solr search index – disabled by default
- `Sitecore.Services.Client.FXM.Enabler.config` – the configuration settings that control the access that an administrator's PC must have to manage the FXM SPEAK application through a website session – disabled by default
- `Sitecore.FXM.Speak.config` – the configuration settings for the FXM SPEAK application

Configuring a distributed environment

In a distributed environment, there is no need to configure SPEAK or search for FXM on the content delivery, reporting, or processing servers.

In a distributed environment, you only need two of the configuration files:

- `Sitecore.FXM.Bundle.config`
- `Sitecore.FXM.config`

On each content management server in a distributed environment, you must specify which content delivery server it should use. In the `Sitecore.FXM.config` file, in the `FXMHostname` setting, specify the content delivery server:

```
<setting name="FXM.Hostname" value="scalingcdservername" />
```

This parameter is empty by default.

Important

To ensure that external pages are tracked, on each content delivery server, you must disable the `Sitecore.ExperienceAnalytics.dll` and `Sitecore.ExperienceAnalytics.Client.dll` files.

Content management – service access and security

Actions that are triggered by the FXM SPEAK application use a `Sitecore.Services` web service. Every request to this service – other than localhost – is disabled by default because you can use a web service to create and delete content items and the service could therefore be used in a malicious attack.

When you set up an FXM content management instance, you must enable the `Sitecore.Services.Client.FXM.Enabler.config.disabled` configuration file by removing the `.disabled` suffix.

If you enable this configuration file:

- Content authoring actions are accepted by the service. In a production environment, you must ensure that your content management server is behind a firewall or that you can manage access to this service with firewall rules.
- You allow cross-domain requests that can access the web service from domains that are different from the domain that the service is hosted on.

  The security policy of the `Sitecore.Services.Client` only permits requests that originate from the localhost by default.

  In a production environment, these requests do not come from localhost; they come from the web browser that is managing the FXM SPEAK application.

The `Sitecore.Services.Client.FXM.Enabler.config.disabled` configuration file is disabled by default. You should only enable this configuration file on content authoring servers that need to make requests to the service in a production environment.

If this configuration file is disabled, you will receive *403 Forbidden* errors every time an administrator tries to create or edit an item in the FXM SPEAK application (such as creating a domain matcher item).

Logging

FXM has its own dedicated log file that is stored in the default log directory. If you require more detailed output in the `web.config` file, set the log level to DEBUG:

```
<logger name="Sitecore.FXM.Diagnostics" additivity="false">   <level value="DEBUG" />   <appender-ref ref="FxmLogFileAppender" /> </lo
```

Note

Debug logging generates a number of extra operations and a large log file. We therefore recommend that you only enable debug logging in a production environment for short periods.

Send feedback about the documentation to docsite@sitecore.net.

FXM JavaScript API

Sitecore Federate Experience Manager (FXM) offers a JavaScript API that enables you to manage some of the Sitecore analytics functionality on webpages on external websites. To use this API, you must set up the FXM JavaScript beacon on the external website.

trackEvent

Use the trackEvent function to trigger a Sitecore page event from a webpage on the external website:

- To trigger the *Page visited* page event*:*

```
SCBeacon.trackEvent(“Page visited”)
```

- To trigger the page event with additional parameters, such as *data* and *dataKey*:

```
SCBeacon.trackEvent(“Page visited”, { data: “custom data”, dataKey: “custom data key” })
```

trackGoal

Use the trackGoal function to trigger a Sitecore goal from a webpage on an external website:

- To trigger the *Register* goal*:*

```
SCBeacon.trackGoal(“Register”)
```

- To trigger a goal with additional parameters, such as *data* and *dataKey*:

```
SCBeacon.trackGoal(“Register”, { data: “custom data”, dataKey: “custom data key” })
```

Send feedback about the documentation to docsite@sitecore.net.

Requirements and limitations

The Federated Experience Manager (FXM) is an integrated part of Sitecore 8.0 and is enabled by default.

To enable FXM functionality on an external website and track visits to that site:

- You must have installed Sitecore xDB.
- You must have physical access to the website.

  You must be able to deploy the Beacon JavaScript on the website.

- Your Sitecore site and the external site must both have the same type of connection – http or https.
- Your license file controls your ability to use FXM, so you must ensure that the appropriate permission is specified in your license file – `license.xml`.

FXM supports:

- Sitecore xDB Cloud Edition
- Sitecore xDB on-premise

FXM supports these internal Sitecore 8.0 features:

- Experience Profile
- Experience Analytics
- Experience Editor

The following content is deployed externally through the Sitecore content mark-up:

- Renderings
- Sublayouts

Limitations

FXM does not currently support the following functionality on external websites:

- Multivariate (M/V) testing
- Sitecore Web Forms for Marketers (WFFM)
- Path Analyzer
- Workflows
- MVC renderings (supported only on Sitecore 8.0 Update-2 or later)

These features are expected to be supported in future versions of Sitecore XP.

FXM has the following limitations:

- FXM does not support external websites that use RequireJS.
- On Internet Explorer 11, to ensure that FXM can process web service requests correctly, the host website must have a top-level domain name.
- The Sitecore website and the external website must use the same application level protocol – http or https.
- You cannot use the same fully qualified domain name (FQDN) for a Sitecore site and a non-Sitecore page or site.

Send feedback about the documentation to docsite@sitecore.net.

Browser and session compatibility in FXM

Because of the differences between browsers, FXM cannot always track contacts with the same ID across multiple sites.

The following tables list the currently supported options:

	CORS support in XMLHttpRequest object	CORS with cookie sending support	Requires P3P headers for cross domain requests	Third party cookie access enabled by default	Session cookies path support
Chrome	y	y	n	y	y
Firefox	y	y	n	y	y
Safari on desktop	y	y	n	n	y
Safari on iOS	y	y	n	n	y
IE 8,9	n (use XDomainRequest with restrictions)	n	y	y	only as sub path
IE 10, 11	y	y	y	y	only as sub path

	External site on same domain	External site on same domain under path	External site on sub domain	External site on separate domain
Chrome	y	y	y	y
Firefox	y	y	y	y
Safari on desktop	y	y	y	n - fallback to unique contact per site with CORS + cookie setting in ext. domain
Safari on iOS	y	y	y	n - fallback to unique contact per site with CORS + cookie setting in ext. domain
IE 8,9	y	y - but not for root page	y (CORS + cookie setting in ext. domain)	n - fallback to unique contact per site with CORS + cookie setting in ext. domain
IE 10,11	y	y - but not for root page	y (CORS + p3p)	y (CORS + p3p)

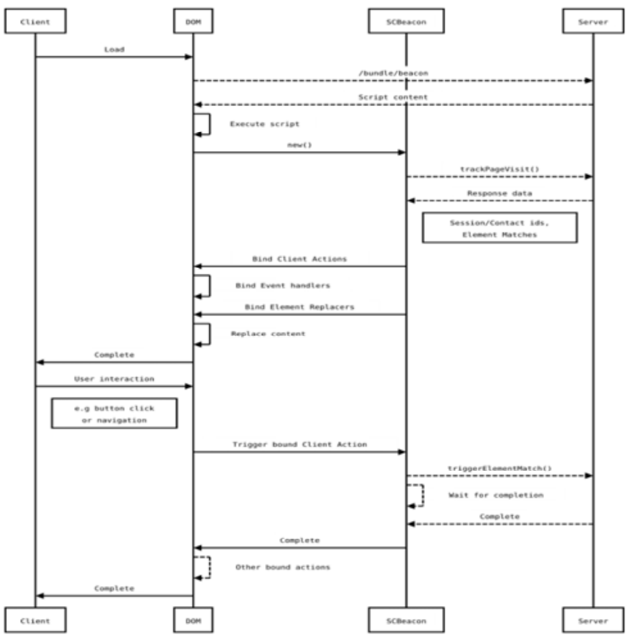
If the external site and the Sitecore site are hosted on different domains:

- For Chrome and Firefox – FXM can successfully track the same contact across both websites.
- For Safari and iOS or any version of IE – FXM assigns each contact a unique contact ID per website.

Send feedback about the documentation to docsite@sitecore.net.

JavaScript-server interaction

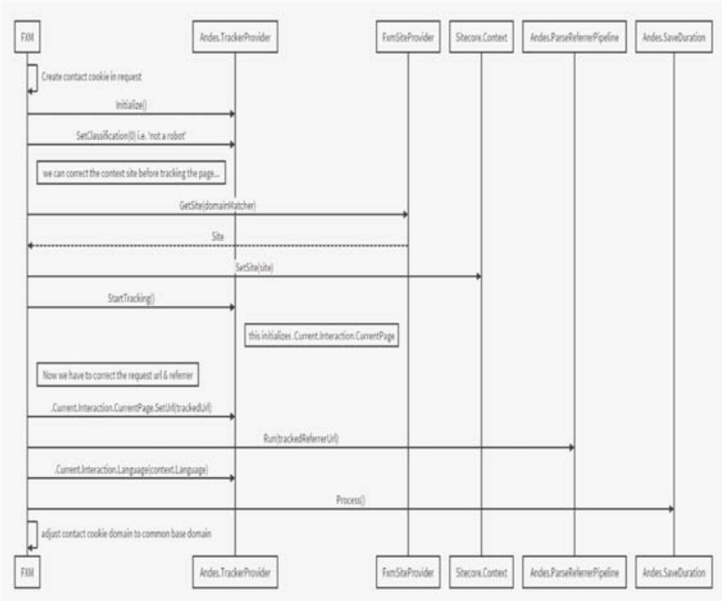
The following diagram describes the process flow of the JavaScript used by Federated Experience Manager (FXM):



Send feedback about the documentation to docsite@sitecore.net.

Page tracking in FXM

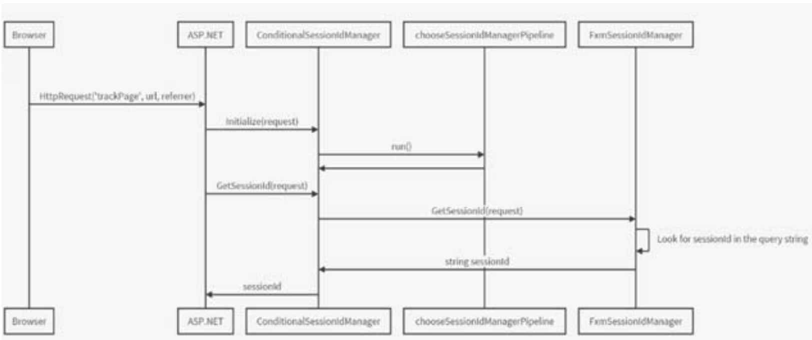
The following diagram describes how the Federated Experience Manager (FXM) tracks a page and how it interacts with the different Experience Analytics pipelines.



Send feedback about the documentation to docsite@sitecore.net.

Session tracking in FXM

The following diagram describes how the Federated Experience Manager (FXM) interacts with the session Manager in Sitecore.



Send feedback about the documentation to docsite@sitecore.net.

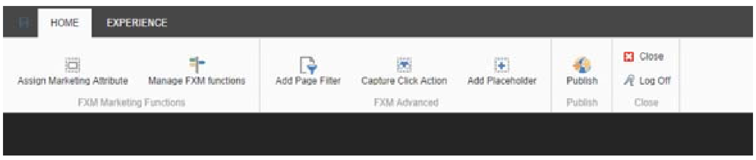
Add a placeholder on an external website in FXM

In FXM, you can use the Add Placeholder function to replace an existing item on an external site with a placeholder. When you have added a placeholder, you can add a Sitecore component to the external site. This enables you to add marketing attributes to the new Sitecore component. The placeholder supports both Sitecore content renderings and sublayouts.

In addition, you can also assign personalization rules with the new component, allowing you to create conditional renderings that you can show to specific groups of visitors.

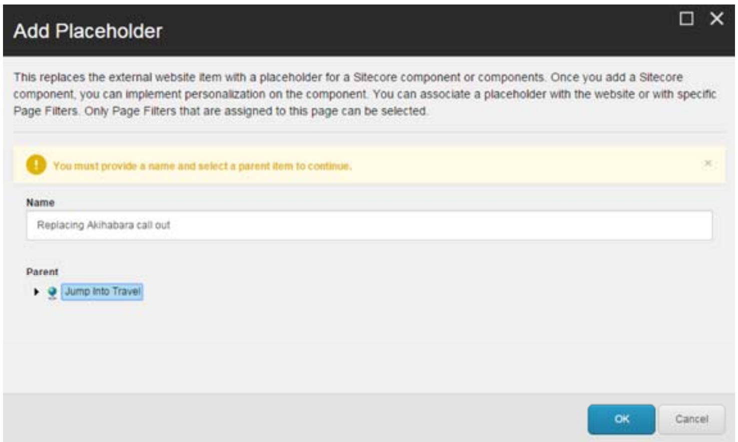
To replace an element on the external site:

1. In the Federated Experience Manager Dashboard, click an external site to select it and then click Open in Experience Editor.
2. On the Home tab, in the FXM Advanced group, click Add Placeholder.



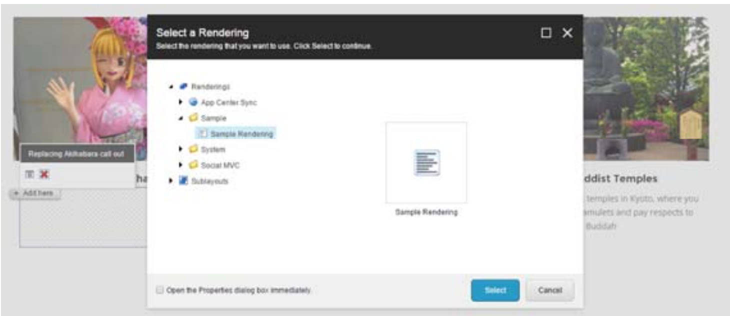
3. On the external webpage, click the component that you want to add a placeholder to and click one of the three options that appear.
    - Add before adds a Sitecore component before the current item.
    - Replace replaces the current component with a Sitecore component.
    - Add after adds a Sitecore component after the current component.

4. In the Add Placeholder dialog, in the Name field, enter a name for the new component, such as *Replacing Akihabara call out*. In the Parent field, click a parent item and then click OK.

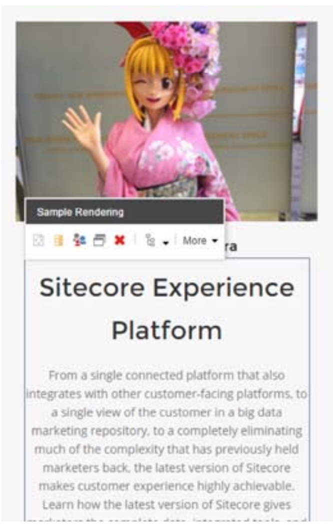


5. In the FXM Experience Editor, on the component that you have just added or replaced, click Add here.

6. In the Select a Rendering dialog, click the appropriate rendering for the placeholder component and then click Select.



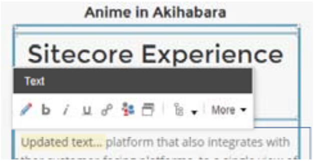
7. On the floating toolbar, click to associate a content item with this component.
8. In the Select the Associated Content dialog, navigate to the relevant item in the content tree and click OK. This replaces the content in the placeholder component's rendering.



9. Click Save and when you are ready to publish the changes, on the Home tab, in the FXM Advanced group, click Publish.

Replace the content in an assigned placeholder

1. To replace the content in the Sitecore Placeholder:
2. In the FXM Experience Editor, on the Home tab, in the FXM Marketing Functions group, click Assign Marketing Attribute. This allows you to edit all the components that you have added a placeholder to.
3. Click the component that you want to edit and add new content to the Sitecore placeholder component. For example, you can replace existing text in the placeholder component with your own text.



Note

You must assign a click action, page filter, or add a placeholder to a component before you can assign a marketing attribute to it. Components that you can assign marketing attributes to are highlighted.

4. Click Save .
5. When you are ready to publish the changes, on the Home tab, in the FXM Advanced group, click Publish.

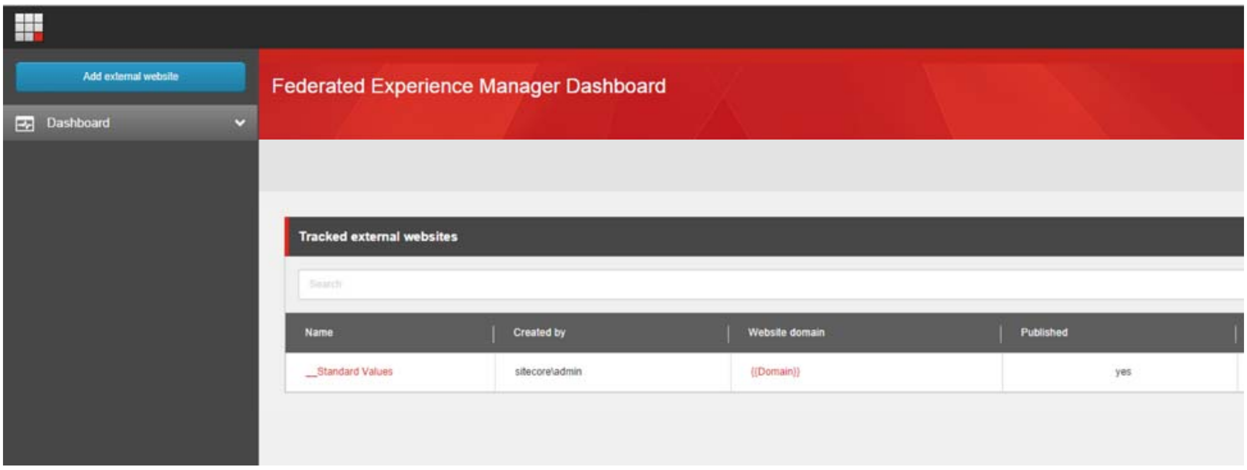
Send feedback about the documentation to docsite@sitecore.net.

Add, edit or delete an external website in the FXM

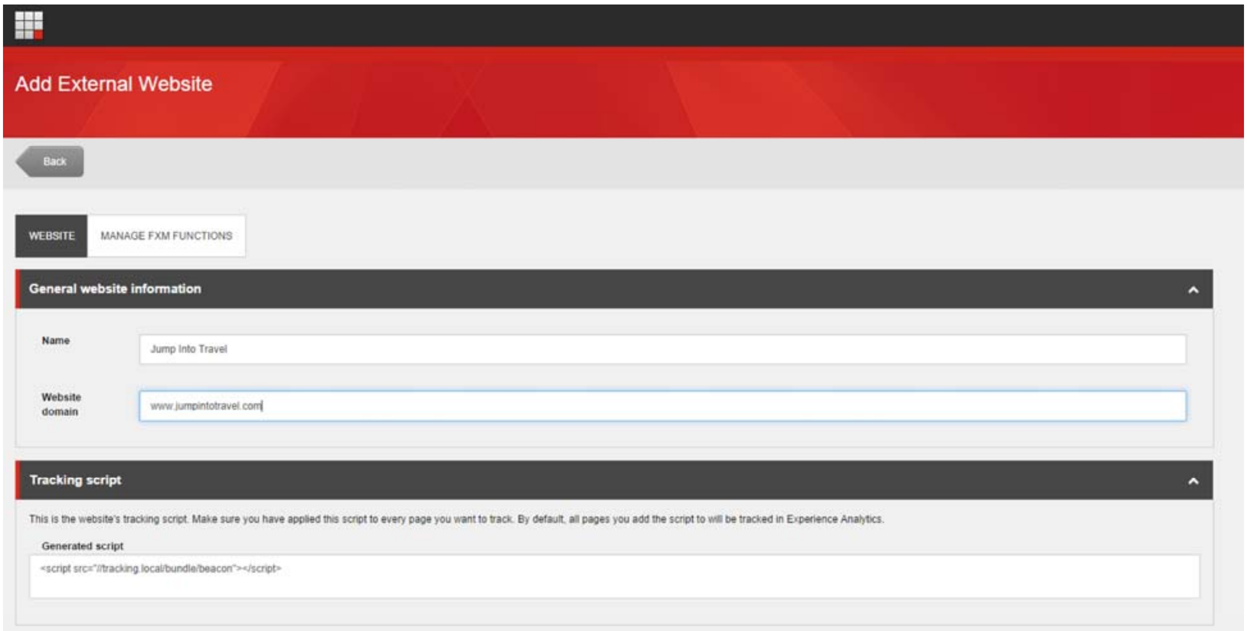
The Federated Experience Manager (FXM) is an application that allows you to track visitor interactions and generate analytics information on external, non-Sitecore websites. You use FXM to implement personalization rules, assign goals and events, as well as implement content profiling on an external site.

To add an external website for tracking in the FXM:

1. From the Sitecore Launch Pad, open the Federated Experience Manager (FXM).
2. On the FXM Dashboard, select Add External Website.



3. On the Website tab, under General website information, in the Name field, enter the website name.
4. In the Website domain field, enter the website's URL.



5. Click Save.
6. On the Edit External Website screen, select Publish to publish the external website. In the Publishing wizard, select Publish.

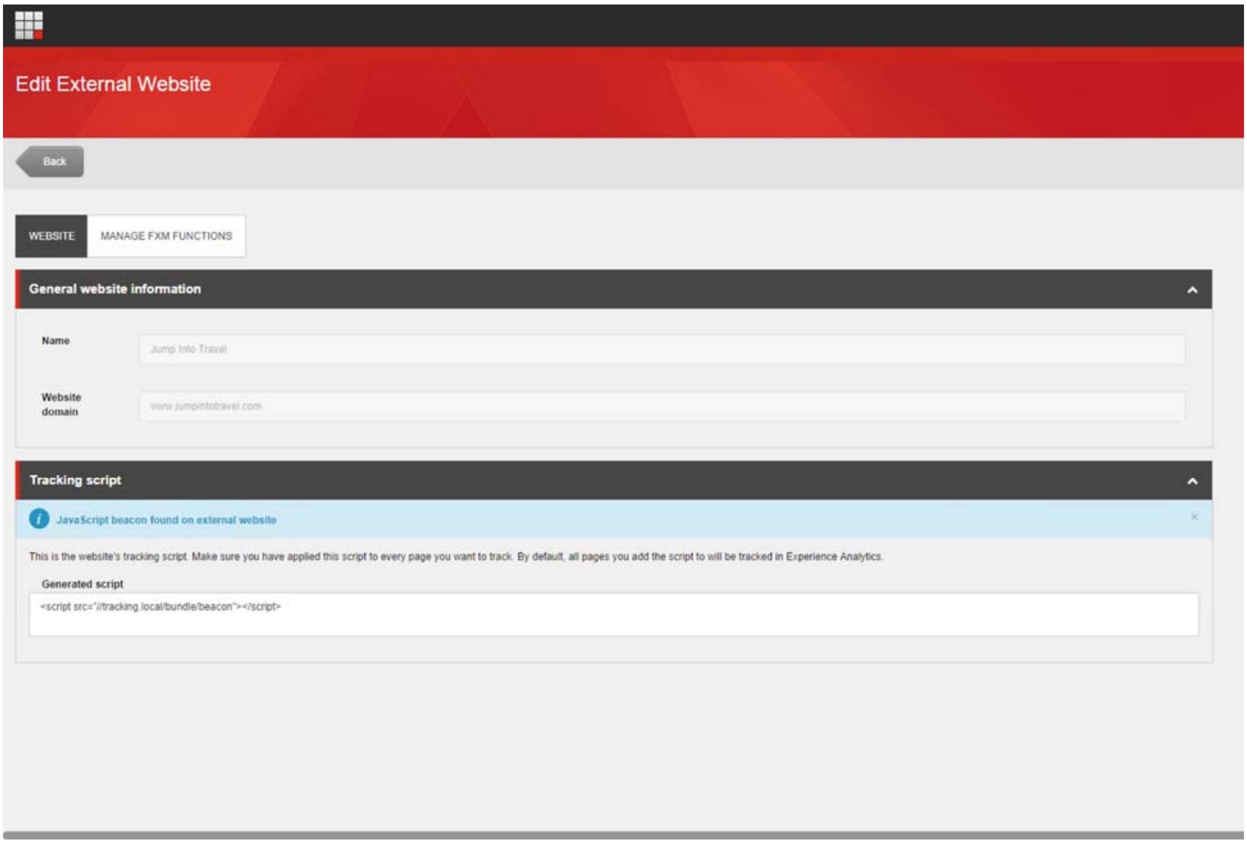
Once you have entered an external website into the FXM, you can edit the website's name or URL.

To edit a website's information in the FXM:

1. From the Sitecore Launch Pad, open the FXM.
2. On the FXM Dashboard, under Tracked external websites, click on the name of website you want to edit. Once you have selected a website, click the edit icon .
3. On the Website tab, under General website information, edit the Name or Website domain fields.
4. Click Save.
5. On the Edit External Website screen, select Publish to publish the external website. In the Publishing wizard, select Publish.

To delete a website from the FXM:

1. From the Sitecore Launch Pad, open the FXM.
2. On the FXM Dashboard, under Tracked external websites, click the name of website you want to delete.
3. On the Edit external website screen, select Delete.



4. Click OK to delete the website.

Send feedback about the documentation to docsite@sitecore.net.

Apply a click action to a component in the FXM

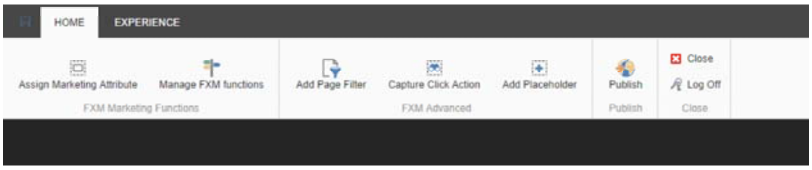
In the Federated Experience Manager (FXM), you can assign Sitecore functionality to components on an external website. You can apply Sitecore functionality on a component level by assigning click actions to external items, such as buttons or links. A click action enables you to track when users click on a particular item on an external website.

Once you have applied a click action to an item, such as a submit button or a link to download a brochure, you can apply marketing attributes such as goals, events, campaigns, and content profiles by using the Assign Marketing Attribute function. By profiling content and adding marketing attributes, you can then track how contacts interact with the external website in Experience Analytics.

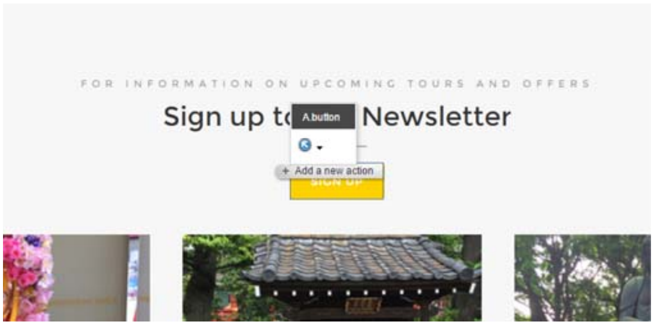
You assign click actions in the FXM Experience Editor.

To assign a click action:

1. From the Sitecore Launchpad, open FXM.
2. On the FXM Dashboard, under Tracked external websites, click the name of website you want to edit and then click Open in Experience Editor.
3. On the Home tab, in the FXM Advanced group, click Capture Click Action.



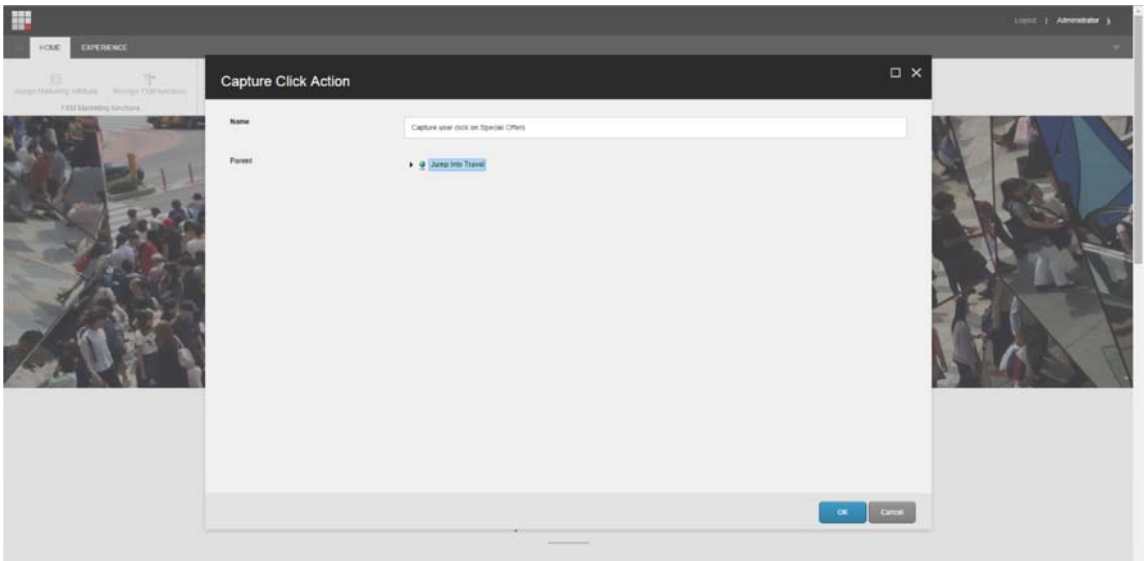
4. On the webpage, click the component that you want to assign a click action to and then on the Add new action button, click +.



5. In the Capture Click Action dialog, in the Name field, enter a name for the click action.
6. Select the parent item where you want the action to be registered, click OK.

Note

You can only select page filters that are assigned to the current page as a parent item.



7. Click Save and then, on the Home tab, FXM Advanced group, click Publish to publish the changes.

Send feedback about the documentation to docsite@sitecore.net.

Apply a page filter to an external website in FXM

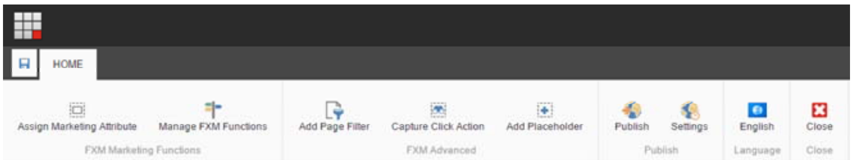
The Federated Experience Manager (FXM) is an application that allows you to track visitor interactions and generate analytics information on external, non-Sitecore websites. You use FXM to implement personalization rules, assign goals and events, as well as implement content profiling on an external site.

You use page filters to assign Sitecore functionality on an individual page or a set of pages of a website, rather than the entire domain.

For example, you can apply a page filter to a page on a website. You can then add a click action to a button on that page that would let you associate an event with the button. When a contact clicks on the button, triggering the event, you can collect and analyze that information in Experience Analytics.

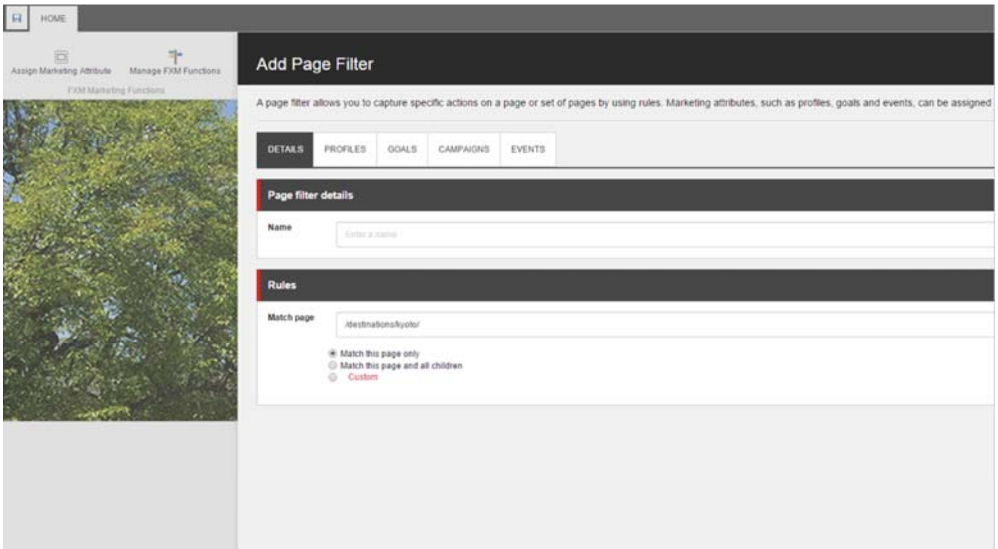
You apply page filters to an external site in the FXM Experience Editor.

1. From the Sitecore Launchpad, open FXM.
2. On the FXM Dashboard, under Tracked external websites, click on the name of website you want to edit. Once the site has been selected, click on the Open in Experience Editor.
3. Navigate through the website to the page you want to assign the page filter to. Once you have found the page you want, on the Home tab, FXM Advanced group, select Add Page Filter.



4. In the Add Page Filter dialog, under Details, in the Name field, enter the name of the page filter. In the same dialog, under Rules, in the Match page field, select the URL of the page you want to apply the page filter to, then select one of the three options from the menu.
   - Match this page only applies the page filter to this page only.
   - Match this page and all children applies the page filter to this page and all child pages.
   - Custom allows you to configure the pages that you which to match via a specific set of rules in the Rule Set Editor.

   Once you have chosen an option, click OK.



   Note

   You can assign marketing attributes to the page filter, such as profiles, goals, campaigns and events, but it is not necessary to do so now. Once you have assigned a page filter, you can add or edit marketing attributes by selecting Manage FXM Functions.

5. Click Save and then, on the Home tab, FXM Advanced group, select Publish to publish the changes.

Send feedback about the documentation to docsite@sitecore.net.

Apply content profiles to an external website in the FXM

The Federated Experience Manager (FXM) is an application that allows you to track visitor interactions and generate analytics information on external, non-Sitecore websites. You can use the FXM to assign profile cards to webpages on the external site.

Profile cards help you apply profile values consistently across the external website. You can also use profile cards to implement personalization rules and create conditional renderings of content on the external site.

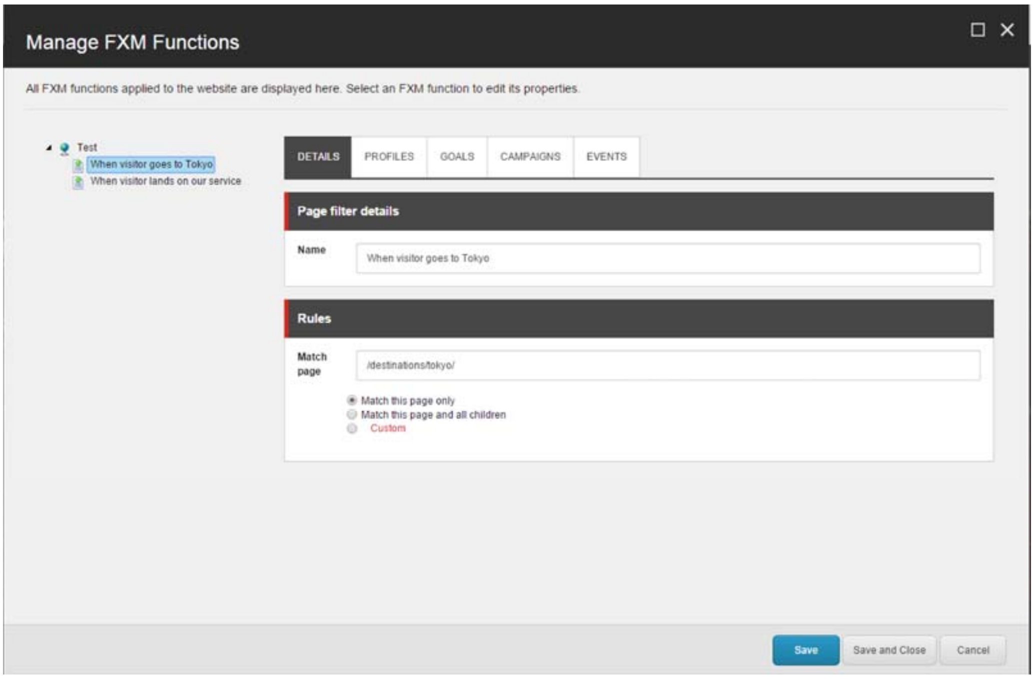
Note

You must create profile cards in the Marketing Center before you assign them to external sites in the FXM.

To apply profile cards to an external website:

1. From the Sitecore Launch Pad, open the FXM.

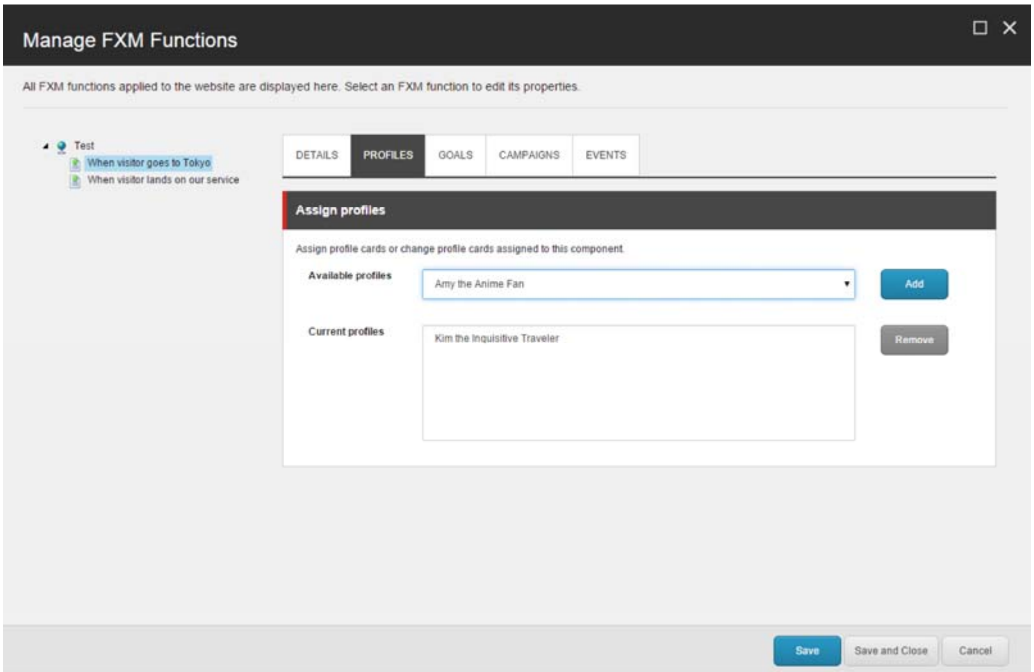
2. On the FXM Dashboard, under Tracked external websites, click on the name of website you want to edit. Once the site has been selected, click on Open in Experience Editor.
3. On the Home tab, in the FXM Marketing Functions group, click Manage FXM Functions.
4. In the Manage FXM Functions dialog, select the component you want to assign the profile to from the list of available page filters, for example, *When a visitor goes to Tokyo*.



Note

You must assign a click action, page filter, or add a placeholder to a component before you can assign a marketing attribute to it. Components that you can assign marketing attributes to have a blue highlight around them.

5. On the Profiles tab, under Assign Profiles, select a profile from *Available profiles*. Click Add to apply the profile, then click Save. You can assign multiple profiles to an external website component.



6. Click Save and then, on the Home tab, FXM Advanced group, select Publish to publish the changes.

Send feedback about the documentation to docsite@sitecore.net.

Assign a goal to an external website component in FXM

The Federated Experience Manager (FXM) is an application that allows you to track visitor interactions and generate analytics information on external, non-Sitecore websites. You use FXM to implement personalization rules, assign goals and events, as well as implement content profiling on an external site.

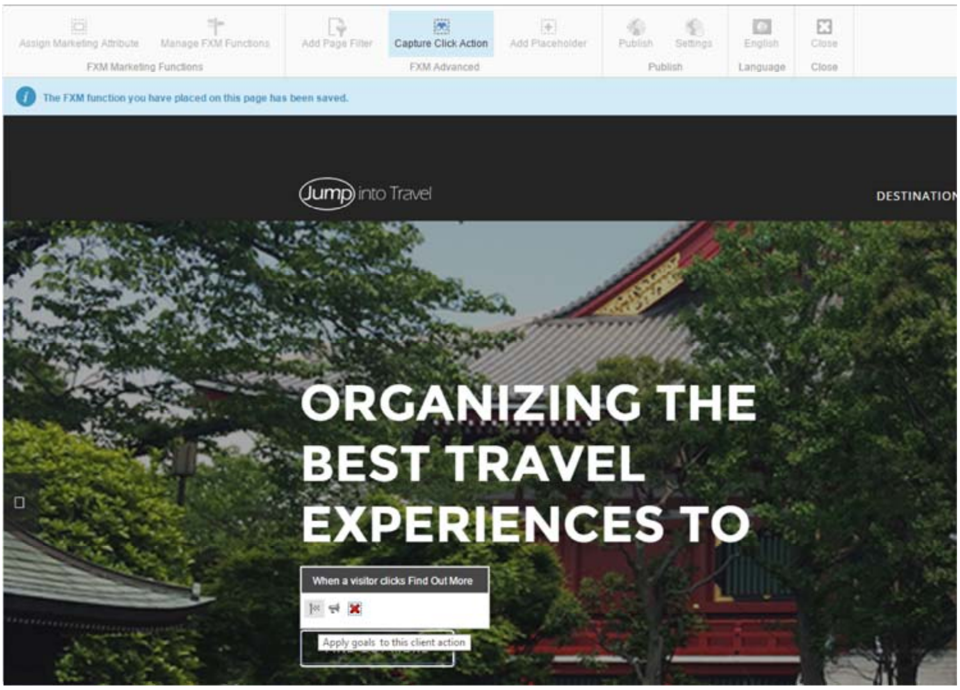
When you have assigned click actions to external website components in the FXM Experience Editor, you can assign marketing attributes, such as goals or events, to components. Goals are activities that visitors can perform on your website. You create goals to track and measure how visitors engage with the website and campaigns – both online and offline. When a contact achieves a goal, it is called a conversion. You can track conversion rates on goals in Experience Analytics.

Note

You must create and deploy goals in the Marketing Control Panel before you can assign them to components in the FXM.

To assign a goal to a component in the FXM:

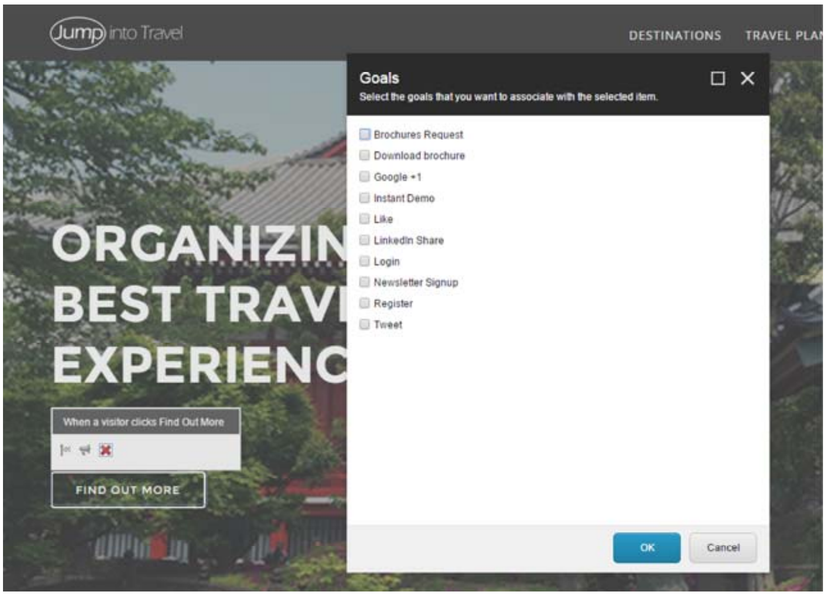
1. From the Sitecore Launchpad, open the FXM.
2. On the FXM Dashboard, under Tracked external websites, click the name of the website you want to edit and then click Open in Experience Editor.
3. On the Home tab, in the FXM Marketing Functions group, click Assign Marketing Attribute.
4. Select the component that you want to assign the goal to, and, on the floating toolbar, select Goals.



Note

You must assign a click action to a component before you can assign a marketing attribute to it. Components that you can assign marketing attributes to have a blue highlight around them.

5. In the Goals dialog, click on the goal that you want to associate with the item, then click OK.



6. Click Save and then, on the Home tab, FXM Advanced group, select Publish to publish the changes.

Send feedback about the documentation to docsite@sitecore.net.

Assign an event to an external website component in the FXM

The Federated Experience Manager (FXM) is an application that allows you to track visitor interactions and generate analytics information on external, non-Sitecore websites. You use FXM to implement personalization rules, assign goals and events, as well as implement content profiling on an external site.

Once you have assigned click actions to external website components in the FXM Experience Editor, you can assign marketing attributes, such as goals or events, to components. Events track visitor activity on a website, which helps build a more complete picture of a visitor's behavior as they navigate your website. You create events, and assign engagement value points to them, in the Marketing Control Panel.

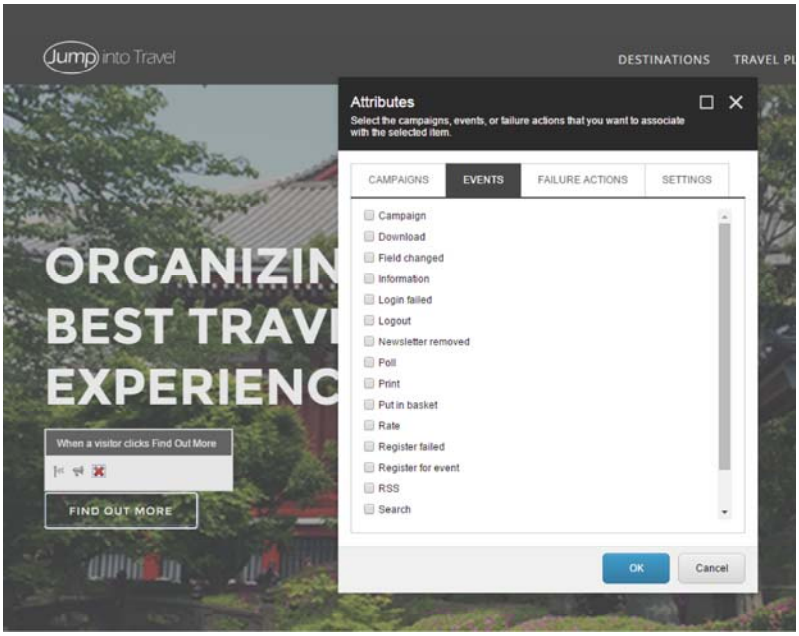
To assign an event to a component in the FXM:

1. From the Sitecore Launchpad, open the FXM.
2. On the FXM Dashboard, under Tracked external websites, click the name of the website you want to edit and then click Open in Experience Editor.
3. On the Home tab, in the FXM Marketing Functions group, click Assign Marketing Attribute.
4. Select the component that you want to assign the event to and then, on the floating toolbar, select Attributes .

   Note

   You must assign a click action to a component before you can assign a marketing attribute to it. Components that you can assign marketing attributes to are highlighted.

5. In the Attributes dialog, on the Events tab, select the event that you want to associate with the component and click OK.



6. Click Save and then, on the Home tab, FXM Advanced group, select Publish to publish your changes.

Send feedback about the documentation to docsite@sitecore.net.

Associate a campaign with an external website in the FXM

The Federated Experience Manager (FXM) is an application that allows you to track visitor interactions and generate analytics information on external, non-Sitecore websites. You use FXM to implement personalization rules, assign goals and events, as well as implement content profiling on an external site.

Once you have applied a page filter to an external website in the FXM Experience Editor, you can associate campaigns with the website or individual pages on the website. Tracking campaigns across your organization's websites can provide you with insight into visitor behavior and help you identify which areas of your website you could optimize to improve conversion rates and visitor engagement. You can track campaign traffic on FXM websites in Engagement Analytics.

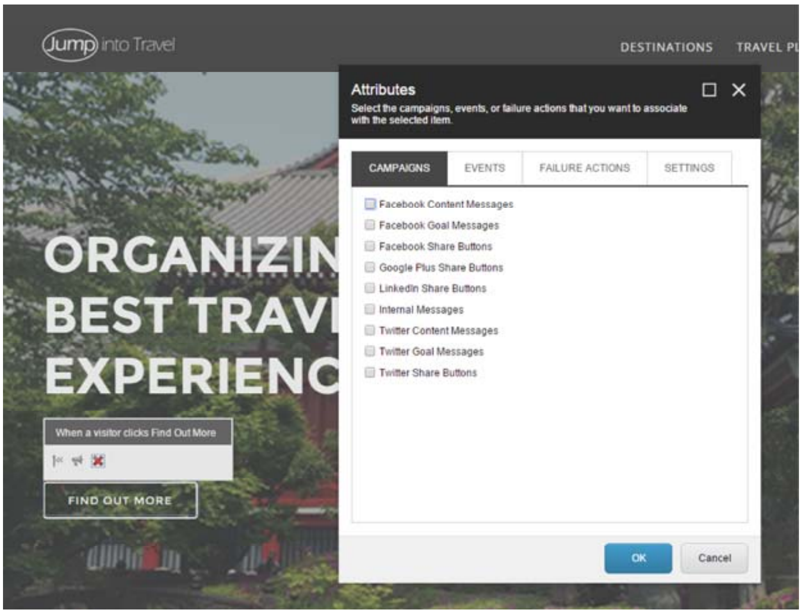
To associate a campaign with an external website in the FXM:

1. From the Sitecore Launch Pad, open the FXM.
2. On the FXM Dashboard, under Tracked external websites, click on the name of website you want to edit. Once the site has been selected, click on Open in Experience Editor.
3. On the Home tab, in the FXM Marketing Functions group, click Assign Marketing Attribute.
4. Select the page or component you want to associate with the campaign, then on the floating toolbar, select the Attributes icon .

   Note

   You must assign a click action to a component before you can assign a marketing attribute to it. Components that you can assign marketing attributes to have a blue highlight around them.

5. In the Attributes dialog, on the Campaigns tab, select the campaign you want to associate with the component and click OK.



6. Click Save and then, on the Home tab, FXM Advanced group, select Publish to publish the changes.

Send feedback about the documentation to docsite@sitecore.net.

Manage FXM functions

The Federated Experience Manager (FXM) is an application that allows you to track visitor interactions and generate analytics information on external, non-Sitecore websites. You use FXM to implement personalization rules, assign goals and events, as well as implement content profiling on an external site.

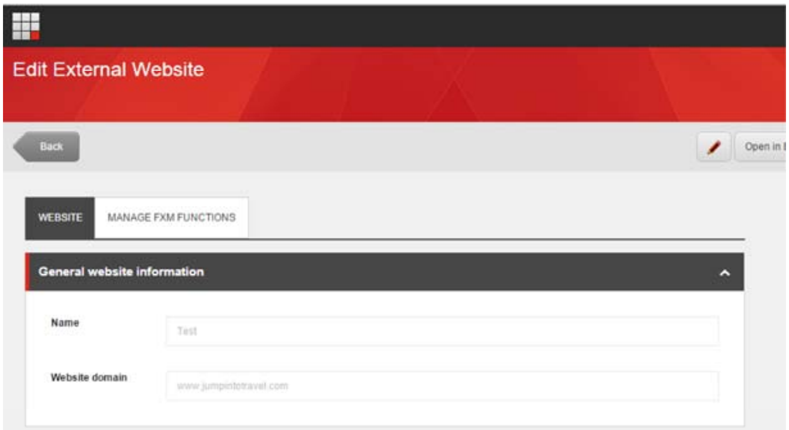
You can view all functions that have been assigned to your external websites using Manage FXM functions. This is available in both the FXM application and the FXM Experience Editor.

In Manage FXM Functions, you can get an overview of all the click actions and page filters that have been applied to an external site. You can edit the marketing function's details, such as its name or the rules associated with the function.

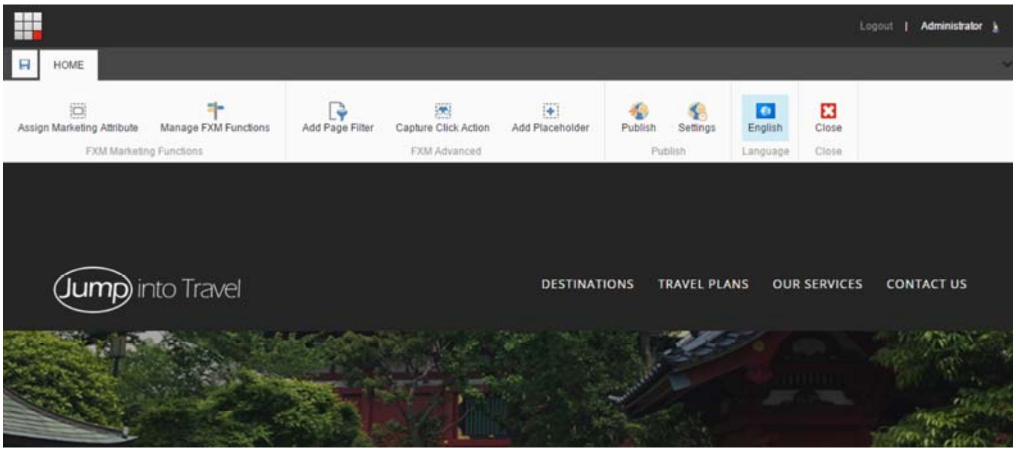
You can view all the marketing attributes of an external website, such as campaigns, goals, events, and profile cards that you have associated with click actions, page filters, and components. You can also edit the associated marketing attributes.

To view all functions that have been assigned to an external site in the FXM:

1. From the Sitecore Launchpad, open the Federated Experience Manager.
2. On the FXM Dashboard, under Tracked external websites, click on the name of website you want to open.
3. To manage the functions assigned to an external site:
   - In the FXM Application, under Edit External Website, click *Manage FXM Functions*.

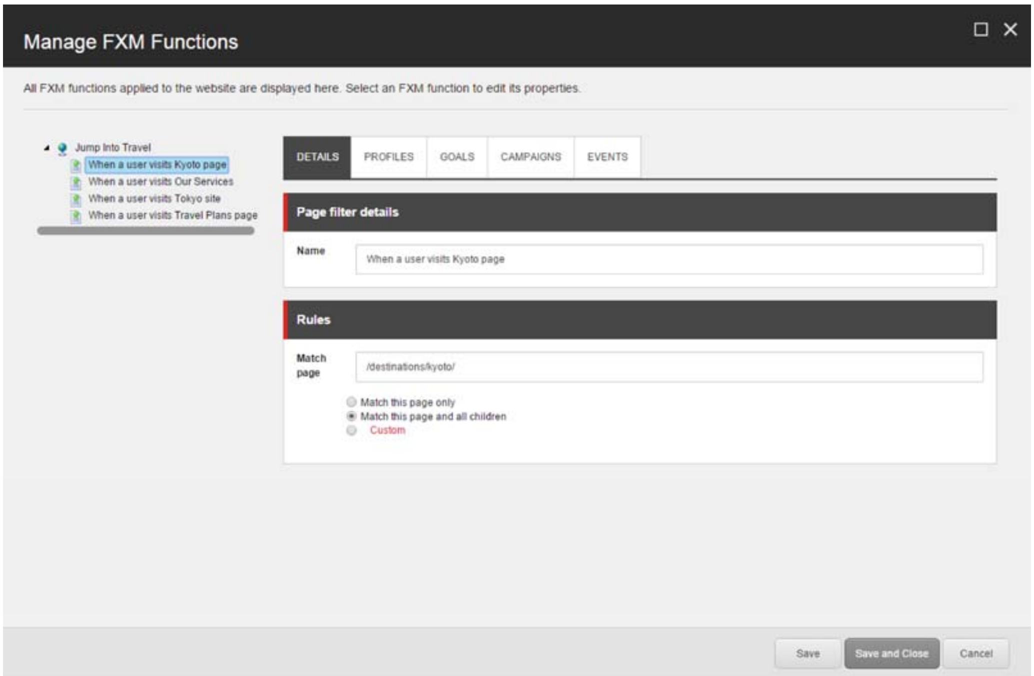


   - In the FXM Experience Editor, on Home tab, in the FXM Marketing Functions group, click *Manage FXM Functions*.



4. In Manage FXM Functions, click the arrow next to external website name to expand the list of assigned FXM functions. To select a specific page filter or click action, click on its name.
5. To select a marketing attribute to view or edit, under Manage FXM Functions, choose a tab by clicking on it. The available attributes are:

Tab	Description
Details	Page filter details field – edit the name of the page filter. Rules field – edit the matching rules for the page filter.
Profiles	Assign Profiles – associate new profiles with the component by selecting them from *Available Profiles* then clicking Add. You can remove profiles from the component by selecting a profile from *Current Profiles* and then clicking Remove.
Goals	Assign Goals – associate new goals with the component by selecting them from *Available Goals* then clicking Add. You can remove goals from the component by selecting a goal from *Current Goals* and then clicking Remove.
Campaigns	Assign Campaigns –add new campaigns to the component by selecting them from *Available Campaigns* then clicking Add. You can delete campaigns from the component by selecting a campaign from *Current Campaigns* and then clicking Remove.
Events	Assign Events – associate new events with the component by selecting them from *Available Events* then clicking Add. You can remove events from the component by selecting an event from *Current Events* and then clicking Remove.



6. If you have added or edited any marketing attributes, click Save.
7. To publish your change, on the Home tab, in the FXM Advanced group, select Publish.

Send feedback about the documentation to docsite@sitecore.net.

Marketing taxonomies

A taxonomy is a hierarchical structure that you can use to identify and organize information. In the Sitecore Experience Platform, you can use taxonomies to classify marketing activities, such as campaigns, goals, and events. You can apply taxonomy tags to these items in the Marketing Control Panel.

The three basic types of marketing taxonomies are:

- Channels
- Campaign groups
- Assets

Marketing taxonomies allow you to track website activities on a more detailed basis, providing you with deeper insight into how contacts interact with your content. This information can help you optimize and refine your marketing strategies.

Using marketing taxonomies

Taxonomy tags provide you with a clear, consistent way to organize your marketing activities. Tagging content with taxonomy tags lets you track how contacts interact with your content. After you have tagged content, you can then use Experience Analytics to measure traffic, engagement value, and value per visit on your campaign groups and individual campaign activities. You can also view reports about how traffic and engagement levels vary across your marketing channels.

You can also use marketing taxonomies to create personalization rules that implement conditional renderings of components based on campaign groups, or the channels a contact uses to visit your website. This helps you to provide contacts with targeted, relevant content that is based on the details that surround their visit to your website.

Create custom taxonomies

Sitecore XP includes channel, campaign, and asset taxonomies, but you can also define custom marketing taxonomies based on your business needs. You can create custom taxonomies that identify different business units, geographical units, individual brands in your organization, and so on. This gives you control over how you track and organize your campaign activities.

You can create custom marketing taxonomies in the Marketing Control Panel and in the Campaign Creator.

Send feedback about the documentation to docsite@sitecore.net.

Classify a media item as a digital marketing asset

When creating marketing campaigns, you may want to track how contacts interact with media items, such as PDFs, videos, or other documents that are on your website. You can do this by classifying media items as digital marketing assets.

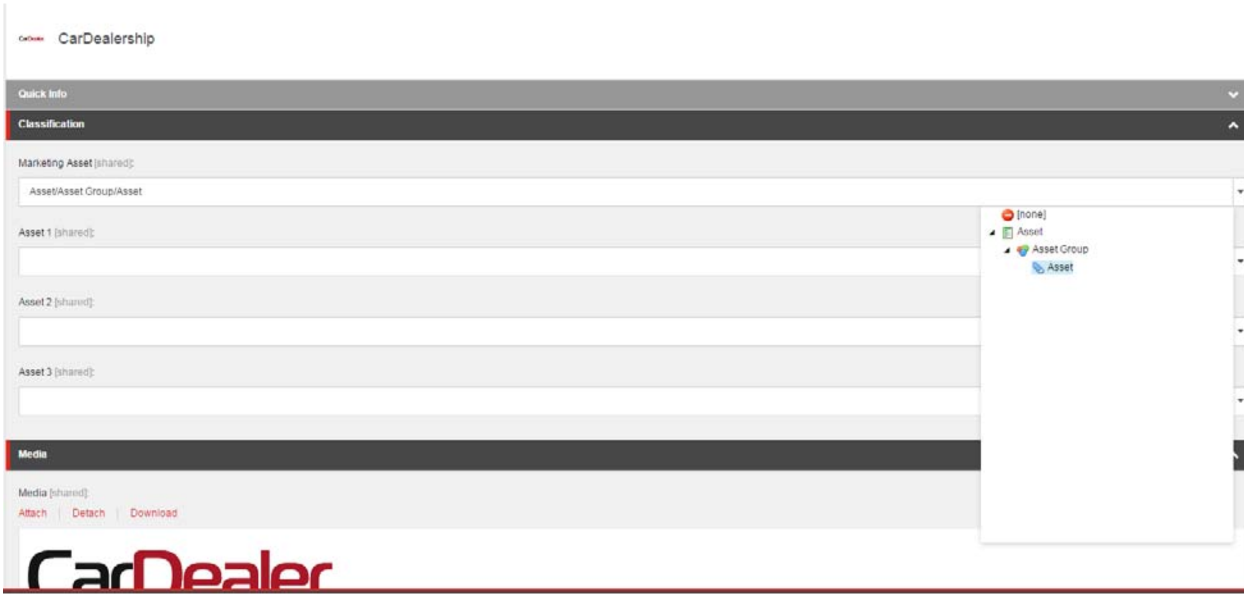
Digital marketing assets are items from the Media Library to which you apply a taxonomy tag. Examples of digital marketing assets are whitepapers, webinars, or other downloadable interactive content.

Once you have applied the taxonomy tag to a digital asset, you can then gather information on how contacts interact with that item. You can use Experience Analytics to track asset downloads and engagement value. You can use the Path Analyzer to view the paths that contacts take to reach the asset.

To classify a media item as a digital marketing asset:

1. On the Sitecore Launchpad, open the Media Library.

2. In the content tree, find the media item that you want to use as an asset.
3. On the Content tab, under *Classification*, select Asset from the drop-down menu. This tags the media item as an asset in the Marketing Control Panel.



4. Click Save and then, on the Media tab, click Deploy.

You can now associate the digital marketing asset with campaigns, events, and goals, and gather analytics information about how contacts interact with the item.

Send feedback about the documentation to docsite@sitecore.net.

Channels

Channels are the paths that contacts use to interact with your brand through campaigns and face-to-face interactions. Contacts can interact with your brand through an app on a mobile phone, by clicking on an ad on a social network to visit your website, or to visit a physical store to purchase an item. With the Sitecore Experience Platform, you can track how your contacts interact with your campaigns across these multiple channels using the Marketing Control Panel.

You can examine how contacts interact with your brand by analyzing campaign traffic levels in the Experience Analytics application. This provides you with insight into how contacts interact with your brand, both online and offline, and which channels provide the highest levels of contact engagement.

You can also use channels to create personalization rules that implement conditional renderings of components based on a contact's channel. This helps you to provide contacts with targeted, relevant content based on the channels that they use to interact with your brand.

Channel types

There are two basic types of channels you use to interact with contacts:

- Online channels – for example, PPC advertisements, a branded mobile application, social media references and organic searches.
- Offline channels – for example, trade shows, direct marketing through the mail, meetings with customers, and conversations with customer service agents over the phone.

You track campaign traffic coming from online channels by associating a campaign with an attributing channel. This could be online activities such as a Twitter tweet, a banner ad, or an e-mail campaign.

Offline interactions can take place in a variety of venues - in specific stores and restaurants, as well as over the phone. You can track these interactions in Sitecore through integration with CRM software, through iBeacon technology, or by using referring URLs connected to your website.

The Marketing Control Panel includes many different types of online and offline channels to help you identify incoming traffic to your campaigns. You can also create your own channel groups and types, which can help you better address your organization's specific marketing needs.

Channels and Engagement Analytics

By comparing channel traffic, goal conversion rates and visitor engagement values across channels, you can examine how contacts interact with your website. You can view reports and dashboards which track channels and referring pages in Experience Analytics. This provides you with insight into how contacts interact with your brand. You can then use this information to optimize your marketing campaigns to provide your contacts with the most relevant content.

By comparing campaign goal conversion rates across channels, you can examine how contacts interact with your brand. You can view reports and dashboards that track channel traffic in the Experience Analytics application. This can give you more insight into how contacts interact with your brand, both online and offline, and which channels provide the highest levels of contact engagement.

Create a channel

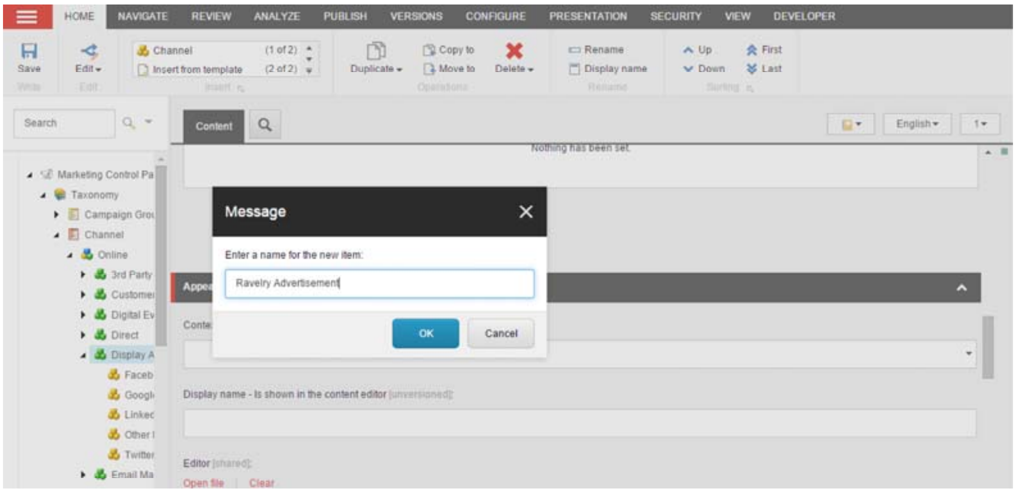
You can track all the interactions that your contacts have with your brand through channels. You can associate channels with campaign activities, allowing you to track how contacts interact with your brand. By comparing goal conversion rates across channels, you can see which channels lead to better contact engagement. You can view reports about channel traffic and referring pages in Experience Analytics.

You can also use channels to create personalization rules that implement conditional renderings of components based on the channel a contact uses to visit your website. This helps you to provide contacts with targeted, relevant content based on the channels that they use to interact with your brand.

The Sitecore Experience Platform includes a number of pre-defined channels, but you can also create your own channels to identify other forms of traffic that are relevant to your company.

To create a new channel:

1. In the Marketing Control Panel, in the content tree, Taxonomy group, select *Channels*.
2. Select the channel type that the new channel belongs to. The channels types are:
   - Offline – for tracking offline interactions
   - Online – for tracking online interactions
3. Select the channel group that the channel will belong to, for example, *Display advertising*. On the Home tab, in the Insert group, select *Channel*.
4. In the dialog box, enter the name of the new channel and click OK.



5. Click Save. To deploy the channel, on the Review tab, in the Workflow group, click *Deploy*.
6. On the Publish tab, in the Publish group, select Publish to perform a site publish.

Map traffic types to channels

In the Sitecore Experience Platform (xDB) version 8.0 and later, the Marketing Taxonomy includes channels that replace traffic types. [Channels]() enable you to classify the different paths that contacts use to interact with your website.

If you have upgraded from an earlier version of Sitecore or have created your own custom traffic types, then you need to map your existing traffic types to channels.

In earlier versions of Sitecore, you can find traffic types in the following location in the content tree:

```
/sitecore/system/Settings/Analytics/Traffic Type
```

In Sitecore 8.0 and later, you can find channels in the following location in the content tree:

```
/sitecore/system/Marketing Control Panel/Taxonomies/Channel
```

Default mapping

The default mapping information from traffic types to channels is defined in the `Sitecore.Analytics.Compatibility.config` configuration file.

```
<compatibility>
<trafficTypeConverter type="Sitecore.Analytics.Data.ConfigurationBasedTrafficTypeConverter, Sitecore.Analytics" />
<trafficType default="{00000000-0000-0000-0000-000000000000}">
  <add trafficType="15" channel="{FB8FA660-0A07-4EE9-A9F4-930FC5D10AEC}" />
  <add trafficType="10" channel="{B979A670-5AAF-4E63-94AC-C3C3E5BFBE84}" />
  <add trafficType="20" channel="{B418E4F2-1013-4B42-A053-B6D4DCA988BF}" />
  <add trafficType="30" channel="{44DD9FF5-44B2-4C59-8DF8-849E400F4B6B}" />
  <add trafficType="31" channel="{44DD9FF5-44B2-4C59-8DF8-849E400F4B6B}" />
  <add trafficType="32" channel="{44DD9FF5-44B2-4C59-8DF8-849E400F4B6B}" />
```

```
  <add trafficType="33" channel="{1DA15267-B0DB-4BE1-B44F-E57C2EEB8A6B}" />
  <add trafficType="34" channel="{1DA15267-B0DB-4BE1-B44F-E57C2EEB8A6B}" />
  <add trafficType="35" channel="{62D19735-E898-4FF5-B53A-13B02D3B8D6F}" />
  <add trafficType="36" channel="{44DD9FF5-44B2-4C59-8DF8-849E400F4B6B}" />
  <add trafficType="40" channel="{B2747066-06F8-4E0B-9EA7-64D8859A119A}" />
  <add trafficType="50" channel="{52B75873-4CE0-4E98-B63A-B535739E6180}" />
  <add trafficType="90" channel="{B5234879-DFFC-47AF-8267-59D4D3DF6226}" />
</trafficType>
</compatibility>
```

This configuration file takes all the standard Sitecore traffic types and converts them automatically to channels. Each traffic type is numbered and maps to a predefined channel that you can identify by the item ID.

In the following example:

```
<add trafficType="15" channel="{FB8FA660-0A07-4EE9-A9F4-930FC5D10AEC}" />
```

- `trafficType="15"` refers to Search Engine - *Organic Branded traffic* type.
- `Channel ="{FB8FA660-0A07-4EE9-A9F4-930FC5D10AEC}"` refers to the *Organic branded search* channel.

Note

The `trafficTypeConverter` node identifies the class that performs the mapping:

```
Sitecore.Analytics.Data.ConfigurationBasedTrafficTypeConverter, Sitecore.Analytics
```

This class is instantiated via the Sitecore object factory and is derived from the `TrafficTypeConverter` class. The implementation specified in the configuration file uses the mappings specified in the configuration file.

The `trafficType` node has a single attribute named `default`. If a traffic type is encountered that has no mapping specified, then this channel ID is returned by default.

Custom mapping

To map your own custom traffic types to channels, you must edit the `Sitecore.Analytics.Compatibility.config` file to include your custom mappings.

Note

Do not make changes directly to the configuration files. Instead, you must create a patch file that performs the required changes during runtime.

To add custom mappings:

1. Click `Website\App_Config\` and in the *Include* folder, open the `Sitecore.Analytics.Compatibility.config` file in an XML editor.
2. Navigate to the `<compatibility>` node.
3. To map a custom traffic type to a channel, create a new entry similar to one of the other entries, with the `trafficType` and `channel` attributes set to your own custom values. For example:

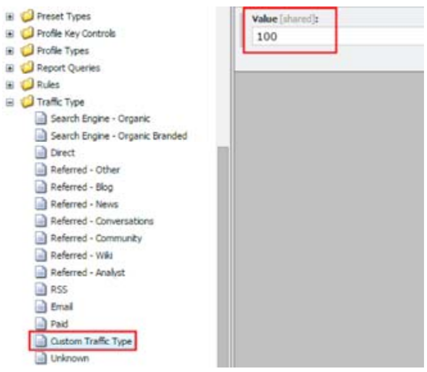
   ```
   <add trafficType="15" channel="{FB8FA660-0A07-4EE9-A9F4-930FC5D10AEC}" />
   ```

4. To find the traffic type parameter, open the Sitecore installation where you created your custom traffic type and navigate to the *Traffic Type* folder:

   ```
   /sitecore/system/Settings/Analytics/Traffic Type/
   ```

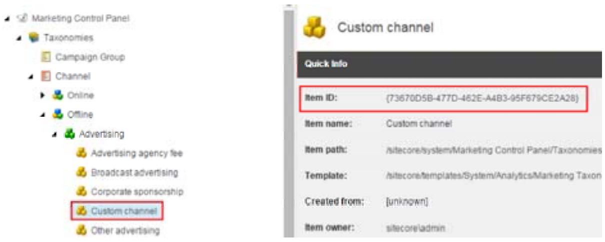
5. Find the *Custom Traffic Type* item that you created and copy the ID from the Value field.

   ```
   trafficType="100"
   ```

6. In the Sitecore installation where you have created your custom channel, open the Content Editor and navigate to the channel node and copy the Item ID of the custom channel item:

   ```
   /sitecore/system/Marketing Control Panel/Taxonomies/Channel
   ```



7. Replace the *trafficType* and *channel* attributes with the IDs you copied, for example:

   `<add trafficType="100"`

   `channel="{73670D5B-477D-462E-A4B3-95F679CE2A28}" />`

   Then save your changes.

8. To test your mapping, create a visit to your website using the traffic type you want to map. Once the session ends or times out the interaction is stored in the xDB.
9. In the xDB MongoDB collection, locate the interaction document for the visit. Verify that the ChannelId field matches the channel ID you have configured in the mapping section.

Note

If you need to create new channels for your custom traffic types, you need to customize the marketing taxonomy.

Send feedback about the documentation to docsite@sitecore.net.

The offline channel groups and their channels

When viewing how contacts interact with your website, it can be helpful to track which channels contacts use to come to your website. This can give you more insight into how contacts interact with your brand, which channels bring the most traffic to your campaigns, and which channels lead to the best goal conversion rates.

Sitecore comes with a number of predefined channel groups and channels. You can also create new channel groups and channels. You can find channels in the Marketing Control Panel.

Offline channel groups and channels

Channel group	Channel
Advertising	Advertising agency fee
	Broadcast advertising
	Corporate sponsorship
	Other advertising
	Out of home advertising
	Print advertising
Call Center	General inquiry
	Other call center
Direct Marketing	Direct mail - print or collateral
	Other direct marketing
Events	Customer events hospitality
	Industry event sponsorship
	Other event
	Trade shows

Partner Marketing	Other partner marketing Partner enablement Partner events Partner lead generation
PR	Other PR PR agency fee PR events
Resources	Corporate and product branding Giveaways Other resources Printed materials Product materials
Sales	Cold calling Customer meeting Inbound phone call Other sales Qualification call Telenurture
Store	Enter store Other store
Telemarketing	Other telemarketing Telemarketing

Send feedback about the documentation to docsite@sitecore.net.

The online channel groups and their channels

When you view how contacts interact with your website, it can be helpful to track the channels that your contacts use to come to your website. This can give you more insight into how contacts interact with your brand, which channels bring the most traffic to your campaigns, and which channels lead to the best goal conversion rates.

Sitecore comes with a number of pre-defined channel groups and channels. You can also create new channel groups and channels. You can find the channels in the Marketing Control Panel.

Online Channel Group	Channel
Organic Search	Search organic branded Search organic non-branded

	Search not provided (Home page)
	Search not provided (not home page)
	Bing ads
	Google ad
Paid search advertising	LinkedIn ad
	YouTube ad
	Other ad
	LinkedIn
	Facebook
Display Advertising	Google Display Network
	Other Display
	PR Referrals
	Affiliate Referrals
Referrals	Partner Referrals
	RSS
	Other Referrals
Email Campaigns	Email Newsletter
	Email Marketing
	Email Notifications
	Email Purchased List
	Email Sent
	Other Email
	SMS Marketing
	SMS Notifications
SMS Campaigns	SMS Sent
	Other SMS
	Email Automation
Automation Programs	SMS Automation
	Other Automation
Apps	App Store Activity

	App Push
	App Usage
	Other Apps
	Blogs
Social Mentions	Wiki
	Other Social Mentions
	Google Plus Social Community
	YouTube Social Community
	LinkedIn Social Community
Social Community	Facebook Social Community
	Twitter Social Community
	Other Social Community
	LinkedIn Sponsored Posts
	Facebook Sponsored Posts
Social Sponsored Posts	Twitter Sponsored Posts
	Other Sponsored Posts
	Webinar Live
Digital Events	Webinar Recorded
	Other Digital Events
	Virtual Events
	Email Inclusion
3rd Party	Other 3rd Party Pay-Per-Lead
	Other 3rd Party

Register a page event programmatically

In the Sitecore Experience Platform, a page event can be triggered by the system or triggered by a visitor to your website. For example, a visitor completing a campaign sign-up form and clicking submit could trigger an event, such as a goal conversion event or a campaign event.

All page event definitions are stored in the following location in the Sitecore content tree:

```
/sitecore/system/Settings/Analytics/Page Events
```

You can configure each page event definition stored here to trigger rules and other actions, and you can associate engagement value points with each page event. In the Content Editor, you can create and configure your own custom page event definitions by adding a new *Page Event* item to the list of existing definitions.

You can register page events on any page or object that implements the `Sitecore.Analytics.Tracking.IPageContext` interface. For example, this could be the current page, the previous page, or any other page or object that implements this interface.

To register a page event:

1. Import the following namespaces into your solution:
   - `Sitecore.Analytics`
   - `Sitecore.Analytics.Data`
2. Use the `RegisterPageEvent` method to register a page event:

   ```
   private void RegisterPageEvent(string name, Guid definitionId, Guid itemId, string data, string text)
   ```

   This method takes the following parameters:

Parameter	Description	Examples
`name`	Name of the page event to be registered.	*Download*, *Search*, *Form Submitted.*
`definitionId`	Unique identifier of the page event.	Item ID of the page event item. `{F358D040-256F-4FC6-B2A1-739ACA2B2983}`
`itemId`	Unique identifier of the page containing the page event.	Item ID of the content item. `{110D559F-DEA5-42EA-9C1C-8A5DF7E70EF9}`
`data`	Custom data or other information related to the page event. For example, in the Experience Profile the *data* field is used to retrieve search keywords for search page events and campaign IDs for campaign page events.	*Search*, *Brochure.pdf* or *7 seconds*.
`text`	Text that describes the page event in a readable format. This text can be an error message or an information message.	*No results found*, *Brochure downloaded*, *Request took 7.263ms to complete.*

3. Include the following code to check that all the necessary preconditions have been met before registering a page event. For example, to check for the current session, the current page, and the interaction related to the page event:

   ```
   Assert.IsNotNull(Tracker.Current, "Tracker.Current");
   Assert.IsNotNull(Tracker.Current.Session, "Tracker.Current.Session");
   var interaction = Tracker.Current.Session.Interaction;
   Assert.IsNotNull(interaction, "Tracker.Current.Session.Interaction");
   Assert.IsNotNull(interaction.CurrentPage,
   "Tracker.Current.Session.Interaction.CurrentPage");
   ```

4. To register the page event, include the following code:

   ```
   var pageEventData = new PageEventData(name, definitionId)
         {
           ItemId = itemId,
           Data = data,
           Text = text
         };
   ```

   The following line ensures that the page event is registered on the current page:

   ```
   interaction.CurrentPage.Register(pageEventData);
   ```

Send feedback about the documentation to docsite@sitecore.net.

Sample Register a page event

After you have created a page event, you also need to register the page event programmatically. The code for registering page events ensures that when the event is triggered by a contact, it is registered in the Sitecore Experience Database (xDB). However, when a page event is triggered it is not stored in the xDB straight away, instead it is stored in the active session. This means that it is only saved to the xDB when the session ends. If the contact is classified as a robot, then the session and the page event are not saved to the xDB.

There are two code samples that you can use to register page events programmatically:

- Standard registration code - full code for registering page events.
- Registration code that prevents duplicate page events.

Standard registration code

This code sample enables you to register page events:

```
private void RegisterPageEvent(string name, Guid definitionId, Guid itemId, string data, string text)
{
  Assert.IsNotNull(Tracker.Current, "Tracker.Current");
  Assert.IsNotNull(Tracker.Current.Session, "Tracker.Current.Session");
  var interaction = Tracker.Current.Session.Interaction;
  Assert.IsNotNull(interaction, "Tracker.Current.Session.Interaction");
  Assert.IsNotNull(interaction.CurrentPage, "Tracker.Current.Session.Interaction.CurrentPage");
  var pageEventData = new PageEventData(name, definitionId)
  {
    ItemId = itemId,
    Data = data,
    Text = text
  };
  interaction.CurrentPage.Register(pageEventData);
}
```

Registration code that prevents duplicate page events

There are two alternate ways to write an *If* statement to check for duplicates, using different LINQ statements.

Using the `All` LINQ extension:

```
if (interaction.CurrentPage.PageEvents.All(pe => pe.PageEventDefinitionId != definitionId))
```

Using the `Any` LINQ extension:

```
if (!interaction.CurrentPage.PageEvents.Any(pe => pe.PageEventDefinitionId == definitionId))
```

If the page event has not already been registered then it can now be registered.

Full code sample to avoid duplicate page events:

```
private void RegisterUniquePageEvent(string name, Guid definitionId, Guid itemId, string data, string text)
{
  Assert.IsNotNull(Tracker.Current, "Tracker.Current");
  Assert.IsNotNull(Tracker.Current.Session, "Tracker.Current.Session");
  var interaction = Tracker.Current.Session.Interaction;
  Assert.IsNotNull(interaction, "Tracker.Current.Session.Interaction");
  Assert.IsNotNull(interaction.CurrentPage, "Tracker.Current.Session.Interaction.CurrentPage");
  if (interaction.CurrentPage.PageEvents.All(pe => pe.PageEventDefinitionId != definitionId))
  {
    var pageEventData = new PageEventData(name, definitionId)
    {
      ItemId = itemId,
      Data = data,
      Text = text
    };
    interaction.CurrentPage.Register(pageEventData);
  }
}
```

Send feedback about the documentation to docsite@sitecore.net.

Create an outcome and an outcome definition

Outcomes describe a visitor's journey from the initial contact to becoming a customer. Through a contact's path to becoming a customer, you can track the events they have triggered and the goals that they have converted on their journey. This provides you with an insight into how contacts interact with your website. You can also assign financial value to outcomes, which allows you to quantify the financial value of a contact for your organization.

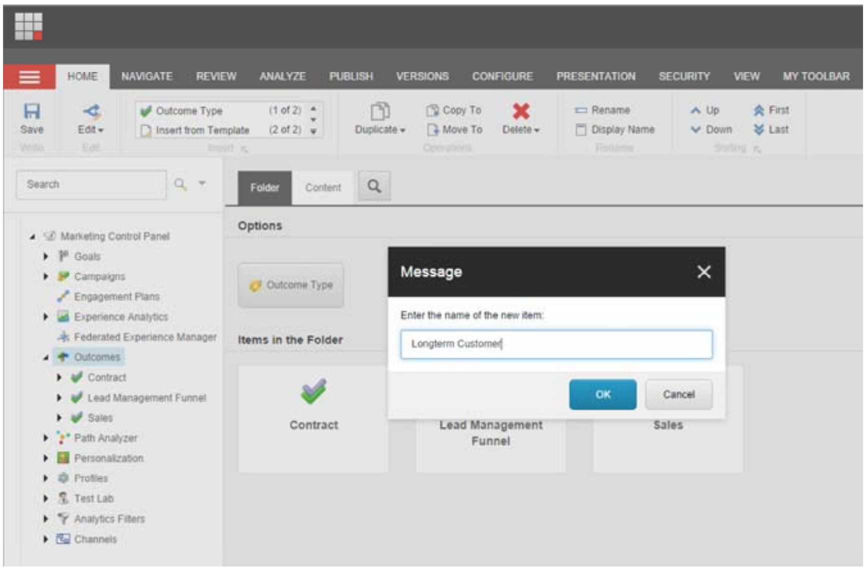
This topic describes how to:

- Create an outcome
- Create an outcome definition

Create an outcome

To create a new outcome:

1. In the Marketing Control Panel, in the content tree, click *Outcomes*.
2. On the Home tab, in the Insert group, click *Outcome Type*.
3. Enter the name of the new outcome in the dialog box and click OK.

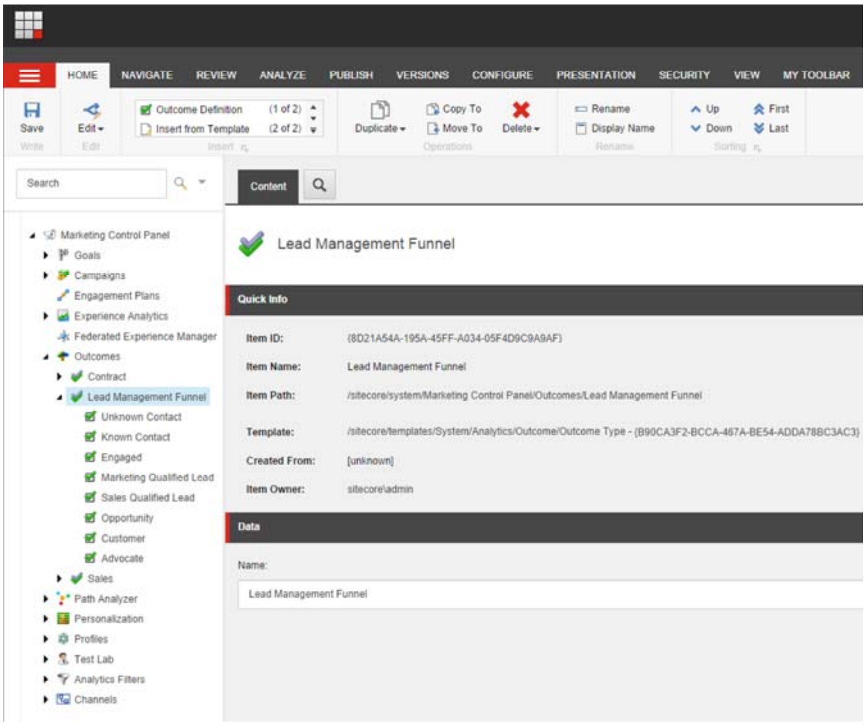


4. On the Home tab, in the Write group, click Save. On the Publish tab, in the Publish group, click Publish to perform a site publish.

Create an outcome definition

To create a new outcome definition:

1. In the Marketing Control Panel, in the content tree, click *Outcomes*.
2. Expand the node of the outcome you want to add a new outcome definition to.
3. On the Home tab, in the Insert group, click *Outcome Definition*.



4. Enter the name of the new outcome definition and click OK.
5. On the Home tab, in the Write group, click Save. On the Publish tab, in the Publish group, click Publish to perform a site publish.

Note

To assign financial value to an outcome, on the Content tab, in the Data section, select Monetary Value Applicable.

Outcomes

An outcome is the business significant result of a dialogue between a contact and a brand. Through a contact's path to becoming a customer, you can track the events they have triggered and the goals they have converted on their journey. This provides you with an insight into how contacts interact with your website, as well as the relative financial value they have for your organization.

With outcomes, you can quantify your contacts based on the financial value they bring to your organization rather than by engagement value . Financial value can be based on monetary value, but you can also define financial value more broadly in terms of the value the customer brings to your organization.

For example, a contact could visit a car dealership website and sign up for a newsletter. The contact visits the dealership, buys the car, and becomes a customer. Three years after buying the car, the customer signs up for an extended warranty plan demonstrating loyalty to the brand, based on the financial value of the purchase. This indicates that the contact may buy other cars from the dealership in the future, providing higher financial value to the organization.

In this example, you can use outcomes to track the financial value of the vehicle sale, as well as identifying the steps in the contact's journey from visitor to lead, and then from lead to customer.

Outcomes and personalization

You can use outcomes to create and implement personalization rules, to provide your contacts with targeted, relevant content. By setting personalization rules that check for defined outcomes, you can ensure that the right content reaches the right visitors in real-time by showing, hiding, or adjusting specific content.

You can also create engagement plans that use outcome types as triggers. Engagement plans help you nurture relationships with your contacts by adapting communication to use the content, channels, and media that are appropriate for each individual interaction.

In the car dealership example, the company could enroll the contact in an engagement plan after signing up for a newsletter. Buying a car would be a trigger that would send them into another engagement plan state for customers. The company could use the Email Experience Manager to send the customer periodic, personalized messages that are relevant to that particular customer. Maintaining a good relationship with customers helps companies retain them, increasing brand loyalty and winning customers for life.

Creating custom outcomes

In the Marketing Control Panel, marketers can create outcome types and definitions. Developers can create and register outcomes programmatically.

In the Marketing Control Panel, outcomes are structured in the following way:

- *Outcome type* - Category containing outcome definition items. For example, *Purchase*.
- *Outcome definition* - A specific outcome item. For example, *Product Purchase* is an outcome definition item under the *Purchase* type.

Registering outcomes

Once you have created an outcome type or an outcome definition item, if you want the outcome to be triggered, it must be registered by a developer. Outcomes are not registered automatically. All outcomes, whether custom or predefined, must be registered programmatically.

An outcome could occur when a contact visits a specific page on a website or it could be triggered in an external system. When an outcome is triggered it is attached to a contact and saved to the xDB collection database.

Predefined outcomes

In the Marketing Control Panel there are three default outcome types. Within each outcome type, there are several predefined outcome definitions:

- Identification – identifies contact acquisition
- Purchase – indicates that a product has been purchased
- Lead Management Funnel – enables you to identify contacts as they progress through your lead management funnel. There are several outcome definitions within the lead management funnel. They are:

Outcome definition	Description
Marketing Qualified Lead	A marketing qualified lead (MQL) who has performed certain actions, such as downloading a white paper or attending a webinar, but is not yet a customer.
Sales Qualified Lead	A sales qualified lead (SQL) who is an interested lead and has the ability to make purchasing decisions. These contacts are typically forwarded to a sales team.
Opportunity	A qualified lead who plans on becoming a customer.
Close won	A closed lead who became a customer.
Close lost	A closed lead who did not become a customer.
Close cancelled	A closed lead who was a customer but has cancelled their contract.

Register an outcome programmatically

As a contact moves from being an anonymous visitor to a known contact and customer, you can track events triggered and goals converted during their journey. Outcomes are business significant events that can be registered in the Sitecore Experience Database (xDB) as a result of one or more interactions between a contact and a brand. They can be either monetary or non-monetary.

Outcomes provide you with an insight into how your contacts interact with your website or external sources, as well as the relative financial value they have for your organization. An example of a website-based outcome could be when a customer buys a product from your website.

In the Sitecore marketing platform, you can act on outcomes by creating personalization, engagement automation, or by analyzing activity using applications such as Experience Analytics and the Experience Profile.

You create and define outcomes in the Marketing Control panel, which also contains several predefined outcome types and definitions. To register any outcome, whether predefined or custom, you must use the API to register the outcome event in the xDB when it is triggered by a visitor on your website. Registering an outcome ensures that the outcome is attached to an existing contact in the xDB.

There are two API classes you can use to register outcomes:

- `TrackerExtensions` - extension methods on the `Tracker` abstract class that enable you to register outcomes in visitor sessions on a Sitecore website.
- `OutcomeManager` - enables you to register outcomes from external data sources and save them directly to your database.

This topic outlines how to:

- Register an outcome during a web session
- Register an outcome from an external source

Register an outcome during a web session

The standard way to register outcomes is to use the `TrackerExtensions` API to register an outcome during a web session.

The `TrackerExtensions` API introduces extension methods on the `Tracker` class that enable you to manage outcomes that take place during a web session.

When you register an outcome using the `TrackerExtensions` API the outcome is not saved directly to the xDB collection database, instead it is stored in the session. This means that while the session is active it is still possible to cancel or modify the outcome, for example, if a contact changes their delivery address during the purchase process. The outcome is only registered and written to the collection database when the session ends.

There are two main advantages of using the `TrackerExtensions` API to register outcomes:

- Performance - the outcome is stored in the session, which means there is less network traffic. When the session ends, the outcome is saved in the xDB.
- Data consistency - the outcome is only stored in the collection database after the contact has been saved to the xDB. This ensures that all outcomes are attached to a corresponding contact. For example, if the contact is a robot, the contact is not saved and therefore neither is the outcome.

To register an outcome during a web session:

1. To use the `TrackerExtensions` API, import the following namespace into your C# source file: `using Sitecore.Analytics.Outcome.Extensions;`
2. Set parameters for the outcome as shown in the following code sample and described in the table below:

```
// Register an outcome in the session. It will only be saved to the database on session end
// and only in case the contact itself is saved.
var outcome = new ContactOutcome(outcomeId, outcomeDefinitionId, contactId);
Tracker.Current.RegisterContactOutcome(outcome);
```

The `ContactOutcome` constructor takes the following parameters:

Parameter	Description
`outcomeId`	The unique ID of the outcome in Sitecore.
`outcomeDefinitionId`	The Item ID of the outcome definition item in the Marketing Control Panel.
`contactId`	The unique ID of the contact stored in the collection database.

1. Use the following code to register an outcome. This method sets the contact ID internally.

```
Tracker.Current.RegisterContactOutcome(outcome);
```

Cancel an outcome (web session)

To cancel an outcome during the current web session use the `DeleteContactOutcome` method as shown in the following code sample:

```
// Unregister an outcome that was previously registered in the current session.
Tracker.Current.DeleteContactOutcome(outcomeId);
```

Get all outcomes

To retrieve all outcomes registered during the current web session use the `Tracker.Current.GetContactOutcomes` method as shown in the following code sample:

```
// Get all outcomes registered during the current session.
IList<IOutcome> outcomes = Tracker.Current.GetContactOutcomes();
```

Register an outcome from an external source

You use the `OutcomeManager` API to register outcomes that are not necessarily website or session related. The `Outcome Manager` API can be used to register all types of outcomes, particularly if they come from other external sources or if they are web sessions from other non-Sitecore websites. It enables you to register and save these outcomes directly to a database without storing them in a session first.

For example, this can be useful if you have an external CRM system or want to collect outcomes registered during offline interactions.

Note

Using the `OutcomeManager` API does not ensure data consistency or improve performance.

When you register an outcome using the `OutcomeManager`, it is saved directly to the xDB collection database. The outcome is still saved even if the contact ID you use does not correspond to a known contact in the xDB.

To register an outcome from an external source:

1. To access an instance of an `OutcomeManager`, use the following factory call:

   ```
   // Instantiate the outcome manager.
   var manager = Factory.CreateObject("outcome/outcomeManager", true) as OutcomeManager;
   ```

2. Use the following code to register an outcome:

   ```
   // Register an outcome.
   var outcome = new ContactOutcome(outcomeId, outcomeDefinitionId, contactId);
   manager.Save(outcome);
   ```

   Note

   You can use a random identifier, such as `ID.NewID`, to generate a random `outcomeId`.

3. The `ContactOutcome` constructor takes the following parameters:

Parameter	Description
`outcomeId`	The unique ID of the outcome in Sitecore.
`outcomeDefinitionId`	The ItemID of the outcome definition item in the Marketing Control Panel.
`contactId`	The unique ID of the contact stored in the collection database.

Get all outcomes

To get all outcomes for a contact, use the following method:

```
// Get all outcomes of a contact.
IReadOnlyCollection<IOutcome> outcomes = manager.GetForEntity<IOutcome>(contactId);
```

Send feedback about the documentation to docsite@sitecore.net.

Personalization

Personalization enables you to deliver targeted, relevant content to your site visitors. By setting personalization rules that check for various details about your visitors, such as location and their visit to your website, you can ensure that the right content reaches the right visitors in real-time by showing, hiding, or adjusting specific content.

You assign personalization rules and conditions to specific items in the Experience Editor.

cms_EE_personalize

Rule-based personalization

Rule-based personalization uses logic-based rules to determine the content that is displayed on a webpage. For example, you can set rules based on the IP address or physical location of your visitors, the keywords they use to reach your site, their mobile device, or the goals they achieve on your website to determine the content that is displayed.

Personalized content is also called a conditional rendering.

Adaptive personalization

Adaptive personalization is a feature that dynamically changes the content of your website based on the visitor's behavior on your website during a visit. Adaptive personalization uses visitor profiles and pattern-card matching to dynamically adapt the content shown to visitors in real-time. You can set adaptive personalization rules in the Rules Set Editor.

Using personalization in testing

You can use A/B and multivariate testing to test your content and then use the results of the test to achieve optimal conversion rates on your site's goals. Testing gives you immediate feedback on what works on your site and what can be improved. You can also use testing to assess which types of content work best with certain segments of your visitors.

Send feedback about the documentation to docsite@sitecore.net.

Ten personalization tactics

To help you get started with personalization, you can download this white paper.

The white paper explains:

- The differences between context marketing and personalization.
- How the Sitecore Experience Platform empowers you to personalize customer experiences.
- Ten proven approaches for personalizing customer experiences.

Send feedback about the documentation to docsite@sitecore.net.

The Rule Set Editor

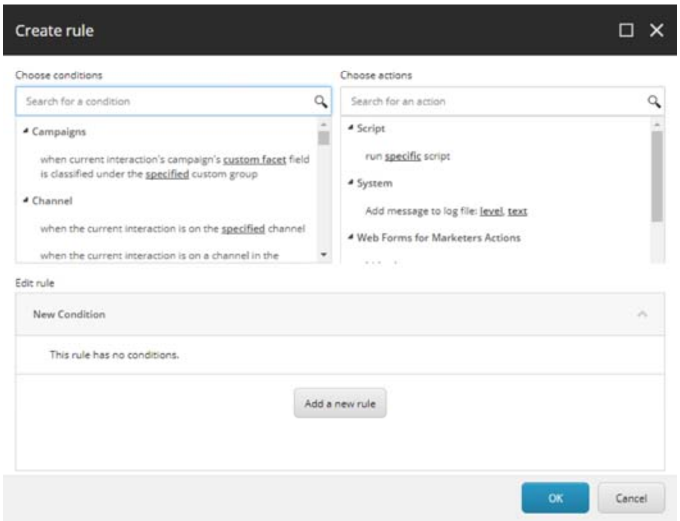
The Rule Set Editor is a tool that applies logic-based rules to content management. You can use the Rule Set Editor to create conditional renderings to personalize and control contacts' experiences.

There are three basic elements in the Rule Set Editor:

- Conditions

- Actions
- Rules

In the Rule Set Editor, you combine conditions with actions to create rules that you can use to implement personalization, run scripts, create steps in engagement plans, and so on.



Conditions

Conditions consist of logical statements that determine whether the condition is true. For example, the where an item is locked by me condition is true if you have locked an item.

Sitecore has a number of default conditions that you can use, but you can also implement your own conditions.

Actions

Actions are the logical steps that are implemented when one or more of the conditions in a rule are true. For example, you can implement a condition for registered contacts that hides a registration form if it's been filled out previously.

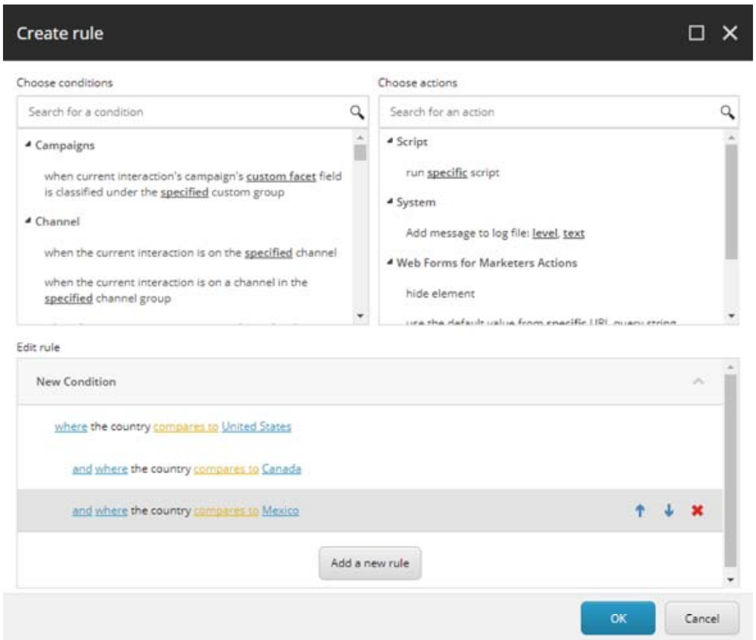
You typically create actions that implement conditional renderings of a website for contacts who meet the criteria of a condition. You can also specify actions that hide or show content if a condition has been met.

Sitecore has a number of default actions available, but you can also implement your own actions.

Rules

Rules associate one or more actions with one or more conditions. You must define your conditions and actions before you can implement a rule. You can also use logical operators such as '*And*' and '*or*' to create combinations of multiple conditions and actions.

For example, you could create a rule that hides a particular content item from contacts from North America. The conditions of your rule check the geographical location of your contacts and if they are from the U.S., Canada, or Mexico, the relevant pages of your website will not be displayed.



Assign a custom profile value to an item

As visitors move through your website, they are assigned the profile values that you have defined for each item that they visit. These values accumulate as the visitor navigates through the site, and they help to build up a visitor profile.

Typically, you use profile cards, or saved sets of profile values, to apply profile values to content. However, some items may not match the profile cards that you have created. In this case, you can assign custom profile values to the item.

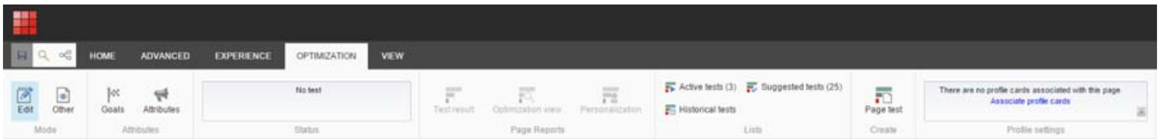
You can also assign custom profile values to multiple items at the same time using the search function in the Content Editor.

Important

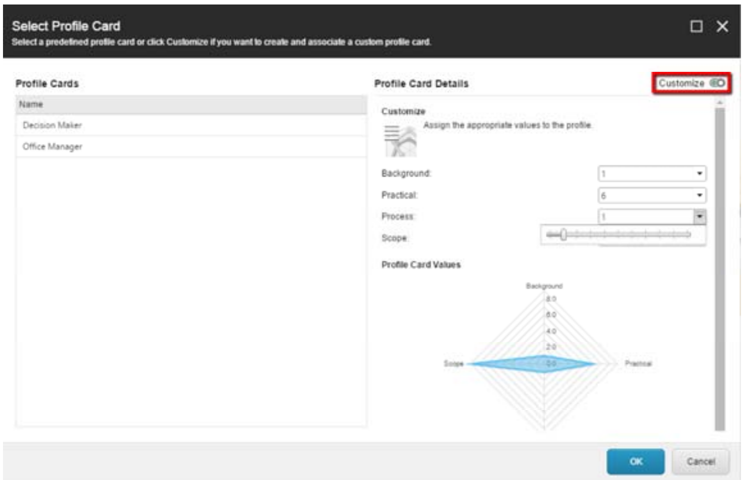
You can only assign custom profile values to items if, in the Select Profile Card dialog, in the Authoring Selection field, you have selected *Single*.

To assign custom profile values to a content item:

1. In the Experience Editor, navigate to the page or content item that you want to assign profile values to.
2. In the Optimization group, in the Page Profile Settings tab, click *Assign Profile Cards*.



3. In the Profile Card Editor, find the profile key that you want to select and click *Edit*.
4. In the Select Profile Card screen, click the Customize switch to enable customization. This enables you to create a custom profile card.



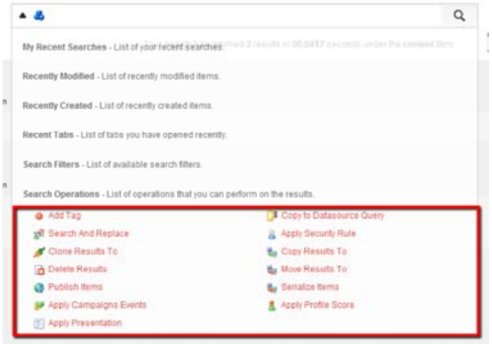
5. In the Customize section, click the drop down arrow for each profile value and select the values that you want to assign to the content item.
6. To save your changes and assign the selected profile values to the content item, click OK.

Assign a custom profile value to multiple items

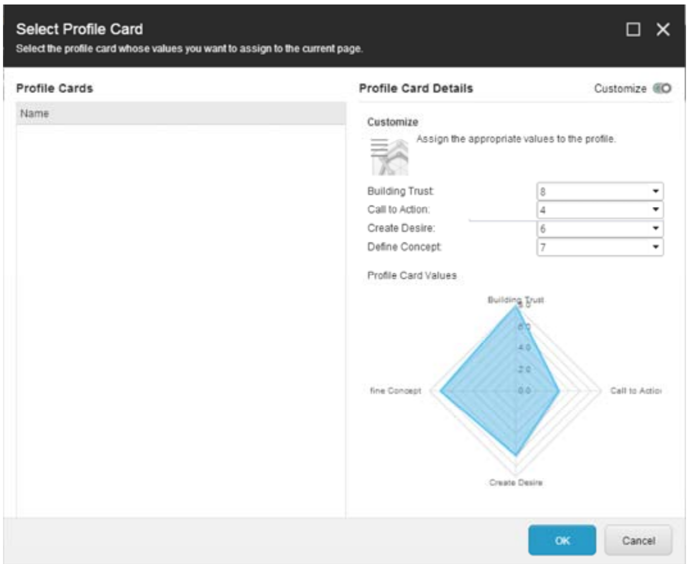
To assign a custom profile value to multiple items:

1. Select an item or item bucket.
2. In the search field enter * to return all items in the item bucket.
3. Click the drop-down arrow to the left of the search field and select Search Operations.

4. In Search Operations, click Apply Profile Score.



5. In the Profile Cards window, in the Customize section, click the drop down arrow for each profile value and select the values that you want to assign to the content items.



6. To save your changes and assign the selected profile values to the content items, click OK.

Send feedback about the documentation to docsite@sitecore.net.

Assign a profile card to an item in the Experience Editor

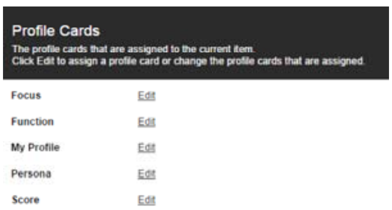
Profile cards save combinations of profile keys and values that you apply to your content. You can use profile cards to assign standardized profile values to items across your website.

You assign profile cards to items in the Experience Editor.

To assign a profile card to an item:

1. In the Experience Editor, navigate to the page that you want to assign a profile card to.

   In the Optimization group, in Page Profile Settings tab, click *Assign Profile Cards*. The Profile Cards dialog box appears.

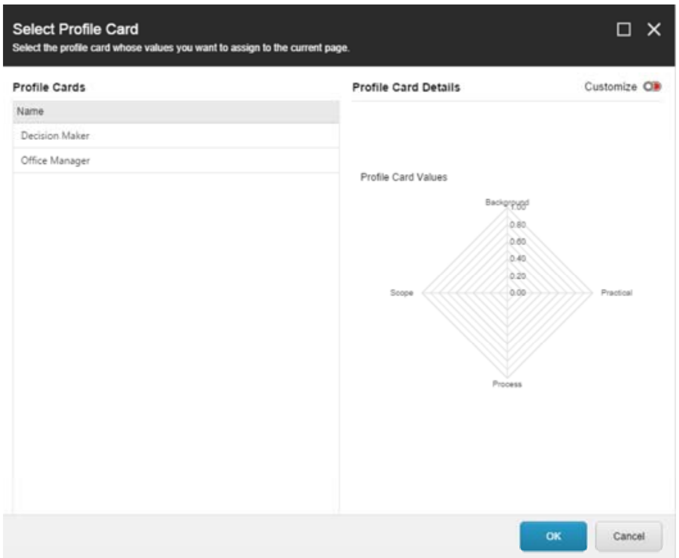


2. In the Profile Cards dialog box, select the profile that you are interested in and click Edit. The Select Profile Card dialog box appears.

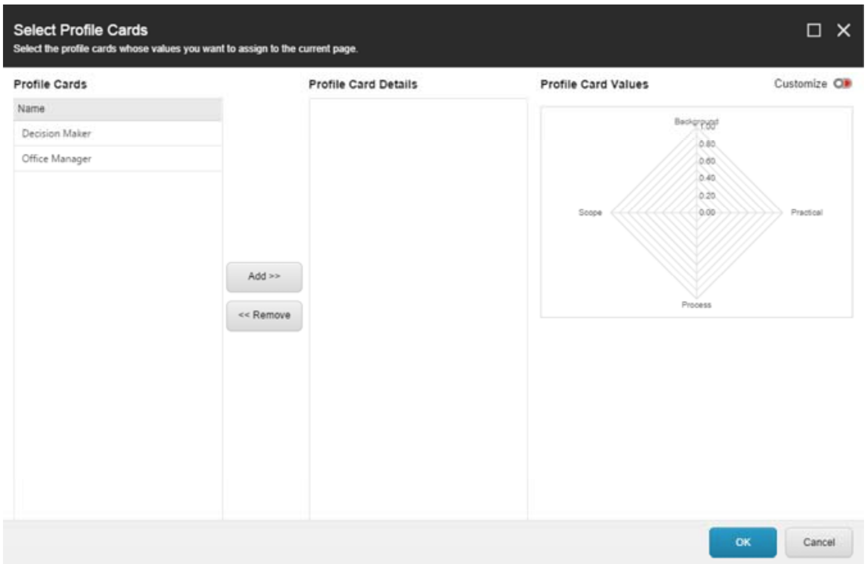
   The Select Profile Card dialog box looks different depending on which option you selected in the *Authoring Selection* field when you created your profile cards. This is located on the Content tab, under Data, *Authoring Selection.*

The options in the *Authoring Selection* field are:

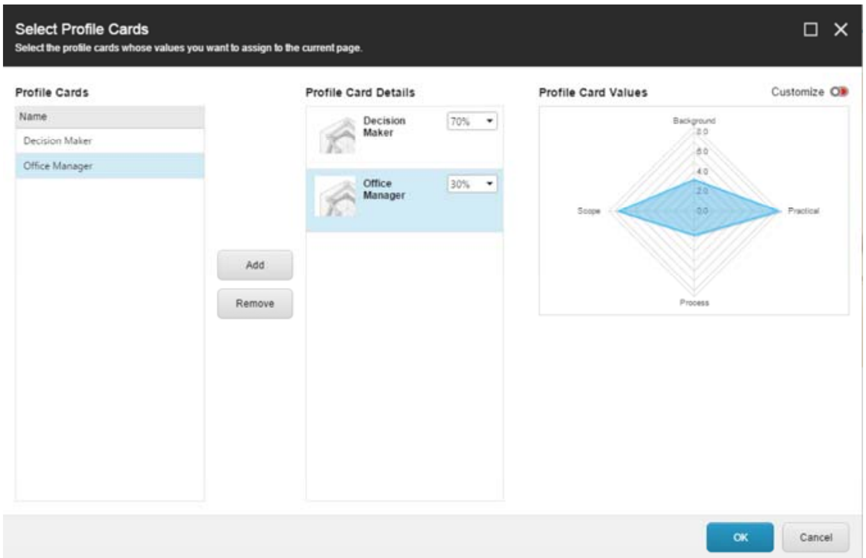
- *Single* - you can assign only one profile card per item. When you choose this option, the Select Profile Card dialog box looks like this:



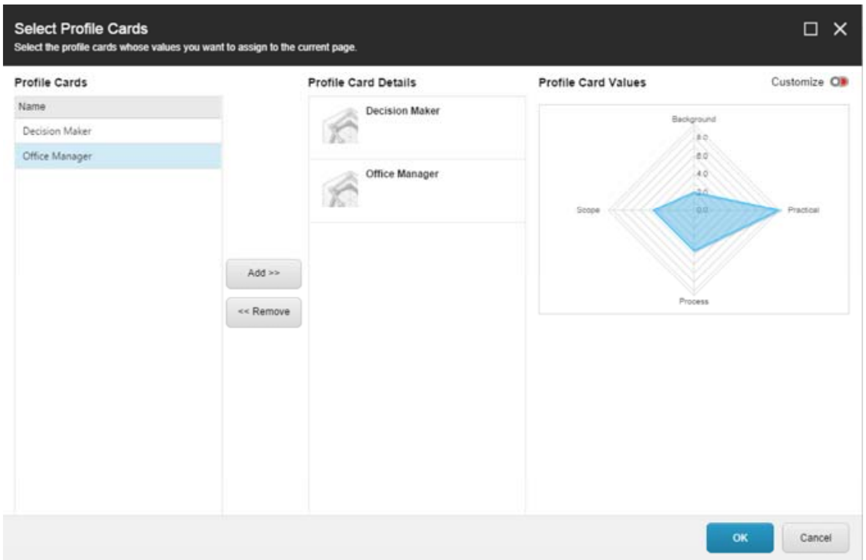
- *Multiple* - you can assign several profile cards per item. When you choose this option, the Select Profile Card dialog box looks like this:



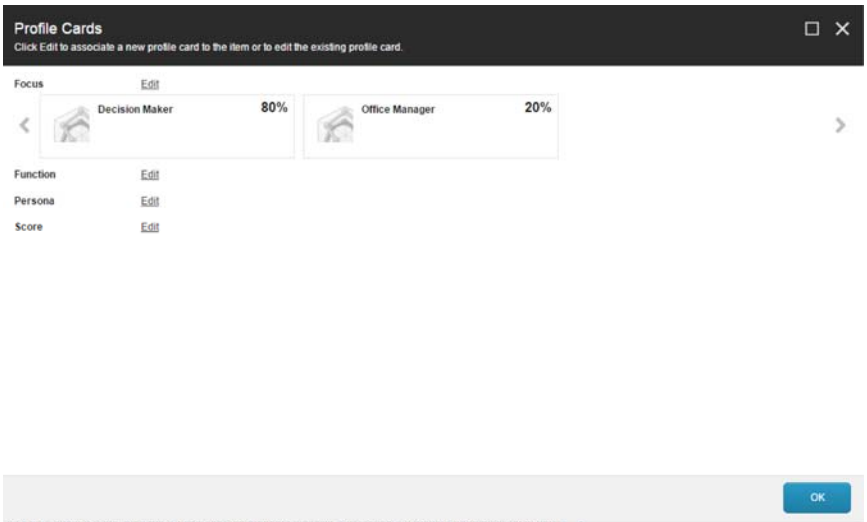
- *Multiple with Percentages* - you can assign several profile cards to each content item and assign percentages to each profile card. When you choose this option, the Select Profile Card dialog box looks like this:



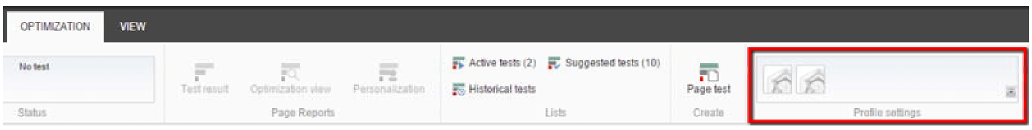
3. In the Select Profile Cards dialog box, in the Profile Cards field, select the profile card that you want to assign to this content item. The profile values that you gave to this card are assigned to the current content item.
   - If you selected the *Multiple* option, select all the profile cards that you want to assign to this content item. When you have finished, click Add.
   - If you selected the *Multiple with Percentages* option, in the drop-down box for each profile card, select the percentage of importance/relevance that you want to assign to each profile card.
   - The sum of the percentages that you select must be 100%.



4. After you have assigned profile cards to the item, click OK. The Profile Cards dialog box reflects the changes you have made.



5. To see the profile cards that you have assigned to a page, in the Experience Editor, on the Optimization tab, in the Profile settings group, move the mouse over the profile card icons. A tooltip with the name of each profile card is displayed.



6. Click OK to save.

Send feedback about the documentation to docsite@sitecore.net.

Conditional renderings

A conditional rendering is a piece of content that is displayed or rendered when a predefined condition is met. You can use conditional renderings to control the way visitors view and interact with your website.

Examples of conditional rendering include:

- Showing different content to visitors based on a rule that checks for their geographic location.
- Hiding a registration form from visitors who previously have filled out the form.
- Changing the text in a website banner based on a visitor's referring site.

Conditional rendering and personalization

Conditional rendering is often used as a synonym for personalized content. It is not, however, a synonym for personalization. Personalization refers to the broad process of delivering targeted, relevant content to visitors.

Personalization includes both adaptive personalization – that is, dynamically changing the content of your website based on the visitor's behavior – and rule-based personalization, which involves the creation and implementation of personalization rules that deliver conditional renderings.

Conditional rendering rules

You set conditions for conditional rendering in the Rule Set Editor. The Rule Set Editor uses logic-based rules to determine if a condition is true. You can define actions, such as a conditional rendering, which take effect when a condition is true, or met. A condition that is linked to an action is called a rule.

You can access the Rule Set Editor from the Experience Editor. You can access it from the Content Editor and Marketing Control Panel as well, but you typically will create conditional renderings in the Experience Editor.

Send feedback about the documentation to docsite@sitecore.net.

Configure pattern matching

Pattern cards are cards that you create to reflect the behavior and interests of specific types of contacts to your website. You assign profile values to pattern cards, then use them to identify different types of typical website activity. You can also use pattern cards to implement personalization rules.

Sitecore uses a measurement called N-Dimensional Euclidean Distance to calculate which pattern card a contact's accumulated profile value most closely resembles and should therefore be mapped to.

When you configure the pattern matching feature, you must specify how many profiled content items a contact must interact with before Sitecore maps their accumulated profile score to a pattern card.

To configure pattern matching:

1. Navigate to `Sitecore build number\Website\App_Config\Include` and open the `Sitecore.Analytics.config` file for your website.
2. In the `configuration/sitecore/settings` section, add the following line:

   ```
   <setting name="Analytics.Patterns.MinimalProfileScoreCount" value="3"/>
   ```

   This means that Sitecore starts to map a contact's behavior to a pattern card when they have requested more than three pages.

3. Change the value to what you think is a suitable number of page requests.

   The value you use should be based on your understanding of your website, its size, and the number of pages that you assign profile values to.

Send feedback about the documentation to docsite@sitecore.net.

Content profiling

Content profiles are categories that you define to track a contact's behavior as they navigate through your website. Content profiling can help you gain a better understanding of the behavior, actions, and interests of your contacts.

Content profiles consist of three basic elements:

- Profile keys – the attributes of the categories you want to track
- Profile values – the numeric values you assign to different profile keys
- Profile cards – saved combinations of profile keys and values that you apply to your content

You can also create personas or fictional characters to represent typical contacts that you assign profile cards to. Personas can be used to implement personalization rules.

You create content profiles, profile keys, profile cards and personas in the Marketing Center. You assign profile values to items in the Experience Editor.

Profile keys

Profile keys describe different aspects of your profiles. You assign numerical profile values to your profile keys, then use profile values to track how contacts interact with the website. Sitecore has some predefined profiles that already have profile keys assigned to them. You can also create your own profile keys.

Profile values

As contacts move through your website, they are assigned the content profile values that you define for each item that they visit. These values accumulate as the contact navigates through the site, and they help you build up a contact profile.

All of the information that you collect about your contacts – the pages they view, the goals they complete, and the path they use to navigate through the site – can help you identify areas of your website that need improvement. You can also use this information to segment contacts and to create rules for personalization.

This information is useful to your sales team. For example, if a contact achieves a high profile value in certain categories, or profile keys, this can indicate that they are a potential sales opportunity. You can enter them into your CRM system as a potential lead, or send them an e-mail.

Profile cards

Profile cards contain saved profile keys and profile values. You can use profile cards to assign standardized profile values to items across your website.

Personas

When setting up profiles for your content, you can create personas. Personas are fictional characters that represent specific contact types within your target demographic. Personas describe the life, age, habits, background, interests, and profession of a fictional character that may use your site in a particular way.

You can create profile cards that describe the way that a persona would consume content on your website.

Send feedback about the documentation to docsite@sitecore.net.

Create a profile key or persona

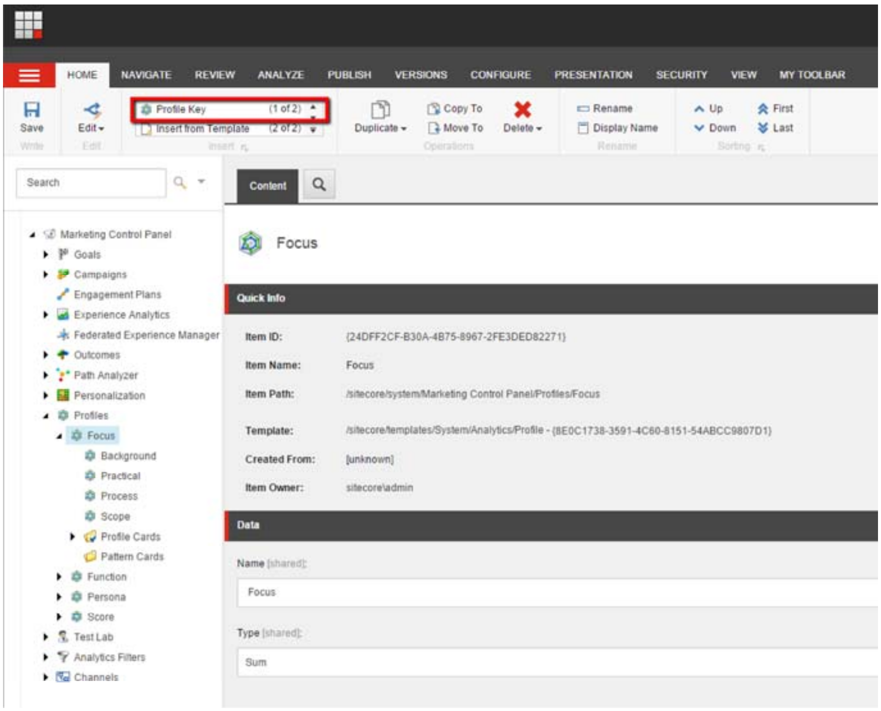
Profile keys describe different aspects of your profiles. Profiles are categories that you define to track a contact's behavior as they navigate through your website. You assign numerical profile values to your profile keys, then use profile values to track how contacts interact with the website.

Personas are profile keys that describe fictional characters that represent specific contact types within your target demographic. You can create profile cards to describe how personas typically interact with content on your website. You can use personas to implement personalization rules.

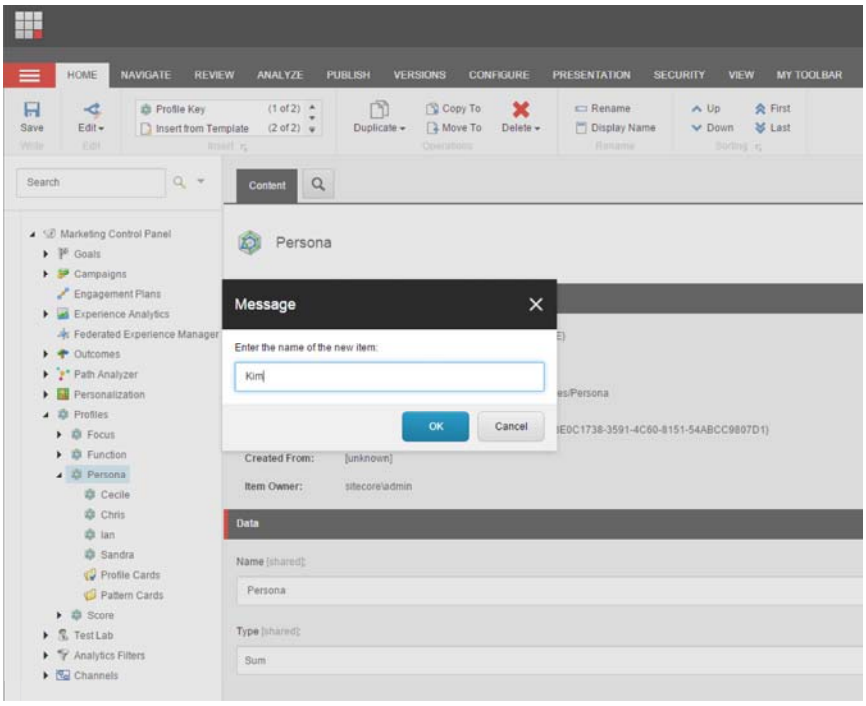
Sitecore has some predefined profiles that already have profile keys assigned to them. You can also create your own profile keys to describe aspects of your website that you would like to track.

To create a profile key or persona:

1. In the Marketing Control Panel, expand the content tree and click *Profiles*.
2. Select the profile you are interested in.
3. On the Home tab, in the Insert group, select the *Profile Key* item.



4. If you want to create a new persona, in the content tree, click *Persona* and then click *Profile Key* and type the name of your persona.
5. Enter a name for the profile key and click OK.



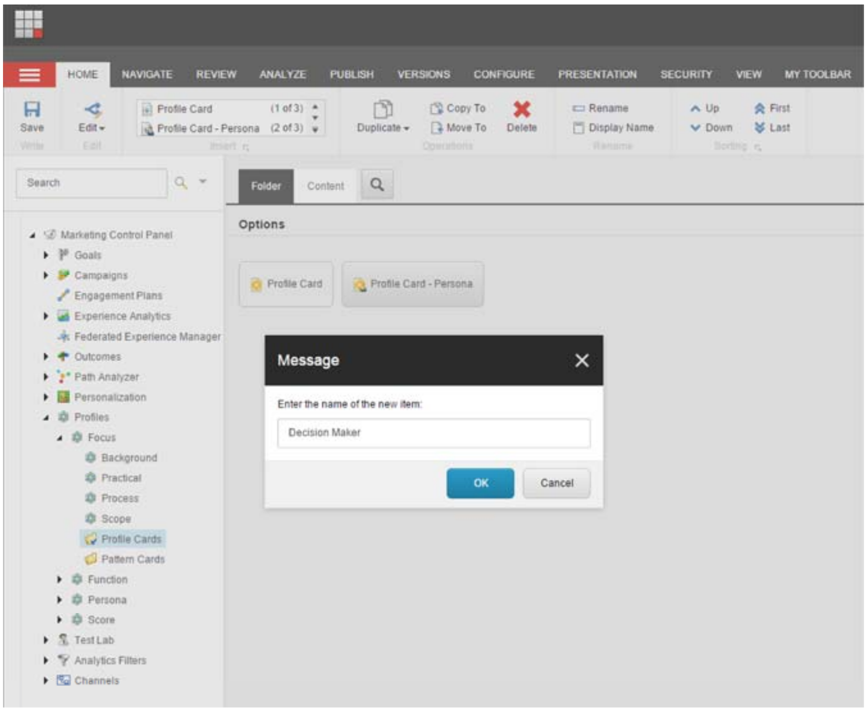
Send feedback about the documentation to docsite@sitecore.net.

Create and assign values to a profile card

You can use profile cards to identify and segment contacts, and implement personalization rules. A profile card is a card that contains saved profile keys and profile values. They allow you to assign profile values consistently across all items on your website.

To create a profile card:

1. In the Sitecore Launchpad, open the Marketing Control Panel.
2. In the Marketing Control Panel, expand the content tree and click *Profiles*. Click on the profile that you are interested in and then click the *Profile Cards* folder.
3. On the Home tab, in the Insert group, click Profile Card to create a new profile card.



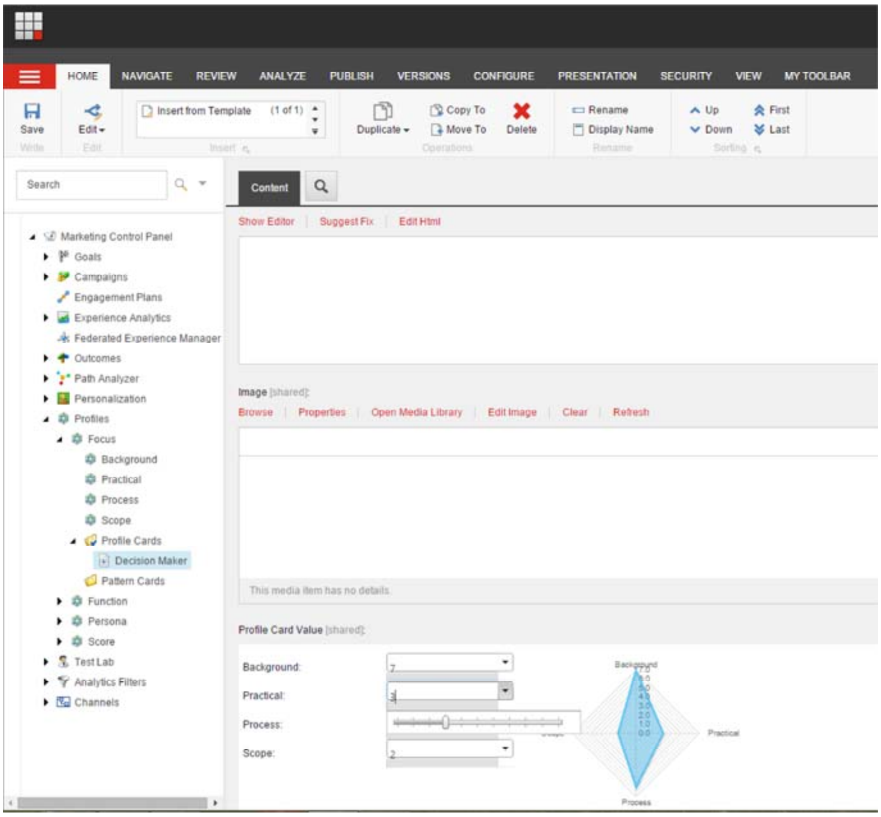
4. Enter a name for the profile card and click OK.

Assigning profile values to a profile card

After you create a profile card, you can assign some relevant profile values to this card.

To assign profile values to a profile card:

1. In the Marketing Control Panel, select the new profile card.
2. On the Content tab, scroll down to the Profile Card Value field.
3. In the Profile Card Value field, select the profile values that you want to assign to this card.



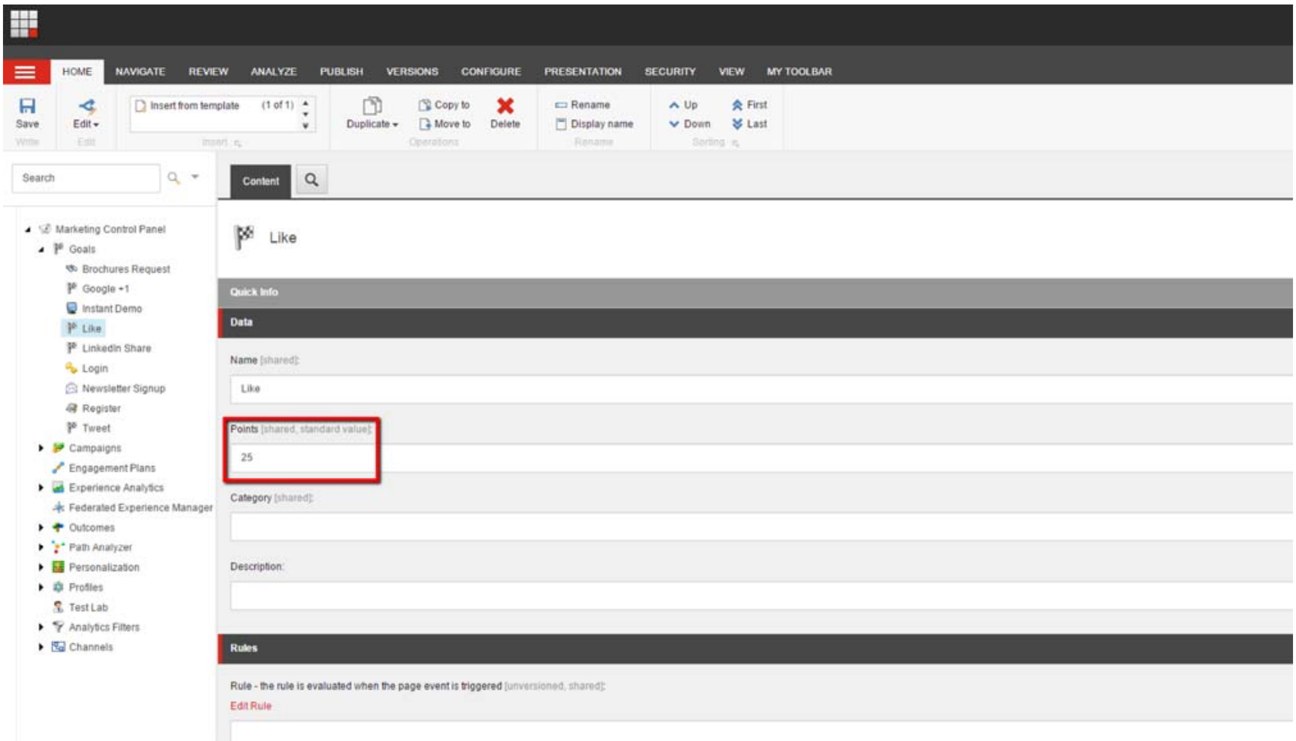
4. Click Save.

Send feedback about the documentation to docsite@sitecore.net.

Engagement Value Points

You assign engagement value points to score specific contact actions and behaviors, such as achieving a goal or triggering a page event. These engagement value points reflect the business value and level of commitment that you associate with a particular contact action. The actual numerical value is less important than the relative value you assign to the event. Engagement value points accumulate across single and multiple contact sessions.

For example, submitting your email address when you register for a newsletter shows a low level of commitment. Requesting a demo shows a higher level of engagement and trust and this contact action should therefore receive higher engagement value points than registering for a newsletter.



Using engagement value points

Engagement value points make it easier for you to measure relative contact engagement across your website. You can use engagement value points to

- Create and implement personalization rules for contacts
- Create engagement plans that evaluate the number of engagement value points that a contact has achieved to determine what action should be taken
- View longer term trends in Experience Analytics, so you can see how effective campaigns are at generating contact engagement

Send feedback about the documentation to docsite@sitecore.net.

Pattern cards

You use pattern cards to map profile values to specific types of visitors and implement personalization rules. These personalization rules enable you to display different content based on your visitors' behavior on your website.

Sitecore compares visitor activity to your pattern cards in real time, allowing you to classify visitors quickly and consistently.

You typically use pattern cards to:

- Implement personalization rules based upon one or more pattern cards
- Define engagement plan steps that take effect when a visitor's behavior matches the pattern card

How pattern cards work

When a visitor navigates through a website, they accumulate the profile values of all the pages and resources that they request. Sitecore calculates the average score that the visitor has accumulated for each profile and maps the visitor to the pattern card that is the closest match.

A visitor can only be mapped to one pattern card per profile at a time. However, as a visitor navigates through your website, the pattern card that they are mapped to can change as they interact with different items and resources.

You can use the personalization rules engine to implement rules based on pattern cards, which will display different content based on which pattern card the visitor best matches.

Send feedback about the documentation to docsite@sitecore.net.

Planning your pattern card strategy

You use pattern cards to map profile values to specific types of website visitors and to implement personalization rules. These personalization rules enable you to display different content based on visitors' behavior on your website.

You also can use pattern cards to define engagement plan steps that take effect when a visitor's behavior matches the pattern card.

You should plan your pattern card implementation carefully to ensure that you create the cards you need. These cards should reflect the specific visitor types or market segments that you are most interested in engaging with.

You should assign appropriate profile values to each pattern card to ensure that profile values are applied uniformly across your website. This can help you to implement personalization rules that deliver targeted, relevant content to visitors whose activities match your pattern cards.

Your pattern card implementation should be based on a careful analysis of your website – its aims, content, and ambitions – as well as an analysis of your target audience – their characteristics, interests, and preferences. This analysis will not only affect the design of your website but also the way you interact with visitors to your website, and the personalization rules that you create.

Send feedback about the documentation to docsite@sitecore.net.

List locking

The List Manager uses the *sitecore_analytics_index* index as a data source. While lists are processed, Sitecore, by default, locks the contact lists and populates the *sitecore_analytics_index* index from the Analytics database.

This means that when you add or remove contacts from a list or clear an entire list, the list locks and you cannot use it in any application, such as EXM, until the processing is complete. This ensures that all contacts in a list are indexed and ready before the list is used. A list unlocks when the unlock list agent has checked and confirmed that all operations have finished.

Note

Refer to the Knowledge Base to troubleshoot list locking issues.

List locking when importing contacts

When you import contacts from a CSV file, the following actions occur in the background:

- A contact list is generated based on the data in the CSV file.
- The expected number of contacts in the import file is saved to the Recipients field on the contact list item.

  Note

  The expected number of contacts is the number of contacts in the CSV file minus the locked contacts or contacts with an invalid identifier.

- The associated list is assigned to the imported contacts.
- The imported contacts are added to the Analytics database in the *Contacts* collection.
- A Mongo entry is created in the contact processing pool. It stores information about a contact and how it should be updated in the Reporting database and the *Analytics* index.
- The imported contacts are updated in the *Analytics* index.
- The unlock list agent (*Sitecore.ListManagement.Analytics.UnlockContactListsAgent*) checks that all jobs are complete and that all the contacts are added to the *Analytics* index, and then it unlocks the list.

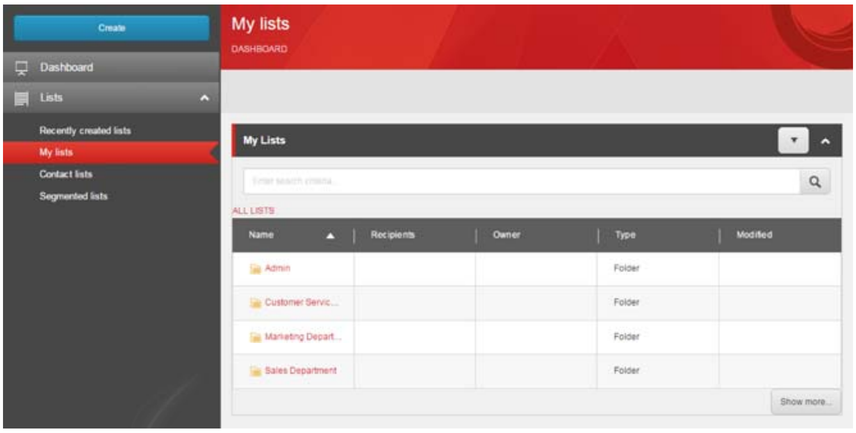
Send feedback about the documentation to docsite@sitecore.net.

Manage lists

When you open the List Manager, the dashboard gives you a quick overview of the lists that are available in Sitecore.

In the List Manager menu, in the List section, you can click a link to see a list of all:

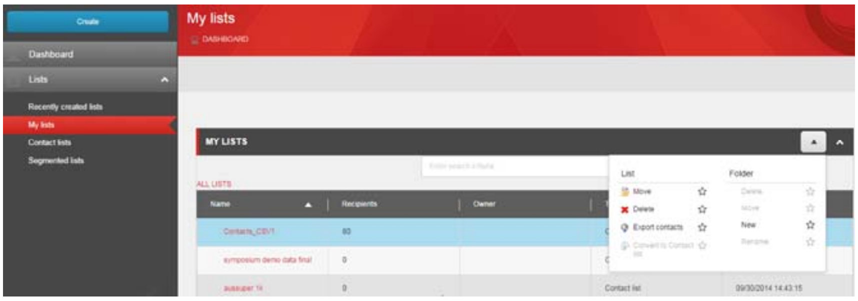
- Recently created lists – displays all recently created lists from the top, regardless of location.
- My lists – displays the lists created by the logged in user or where the logged in user is specified as the owner.
- Contact lists – displays all the Contact lists, regardless of location.
- Segmented lists – displays all the Segmented lists, regardless of location.



From each of these lists, you can click the name of the list to open it. In addition, if you want to delete a list or organize your lists in folders, you can click the list row and then click the drop-down arrow. Depending on the type of list that you click or if you click a folder, different options are enabled in the drop-down menu. The options are:

- List
  - Move - move the list into a different folder.
  - Delete - delete the list.
  - Export contacts - export the contacts that are included in the list to a CSV file.
  - Convert to contact list - convert a Segmented list to a Contact list.
- Folder
  - Delete - delete the folder.

- Move - move the folder into another folder.
- New - create a new folder. If you have clicked a folder, the new folder is created as a subfolder; otherwise, the folder is created in the folder root.
- Rename - rename the folder.



Send feedback about the documentation to docsite@sitecore.net.

The List Manager

In the List Manager, you can manage your Sitecore contacts in lists and create the lists of recipients for your email campaigns.

In the List Manager, you can use the [Rule Set Editor]() to filter the contacts in a list so that it includes only a very specific segment. For example, you can filter the contacts based on their contact information or engagement value points. In this way, in the [Email Experience Manager](), you can create a message and target it to a specific segment of your customers who you think will find your particular message relevant.

You open the List Manager from the Launchpad:

All_LaunchPad_ListManager

List types

In the List Manager, there are two types of lists:

- Contact list – a standard list that contacts can be members of. A contact can be a member of one or more contact lists.

  When you create a new contact list, you can use existing contact lists as building blocks for the new list that together make up a specific target group. For example, you can create a list called North America - Customers, from two existing contact lists: US - Customers and CA - Customers.

- Segmented lists – a list that is a filtered list of one or more selected contact lists. The contacts in a Segmented list are not actually members of the list - they only appear in the list if they are in the list source (the selected contact lists) and they meet the conditions that you define.

  A segmented list updates continually when the contacts in the sources of the list meet the defined conditions.

  You can convert a segmented list to a new contact list. This allows you to take a "snapshot" of the segmented list and create a new contact list with the contacts that meet the defined conditions at the time of conversion.

List source

When you create a list in the List Manager, you can choose to create an empty contact list or you can choose a source of the list:

- For Contact lists, a list source can be any existing contact list.
- For Segmented lists, a list source can be all the Sitecore contacts or an existing contact list.

Note

Only Contact lists can be used as a list source. If you want to use a Segmented list as a list source, you must first convert the Segmented list to a Contact list.

Exclusions

It is also possible to exclude certain contact lists from your sources to modify the list source. This allows you to further shape your target group. For example, you can target the USA customers in a North America list, by excluding a Canada customers list.

You can also use exclusions for quality assurance purposes, for example to exclude unsubscribed customers.

Send feedback about the documentation to docsite@sitecore.net.

Convert a Segmented list to a Contact list

Because the number of contacts in a Segmented list changes continuously depending on the source of the list, you can convert a Segmented list to a Contact list if you want to save and reuse the contacts that are in the Segmented list.

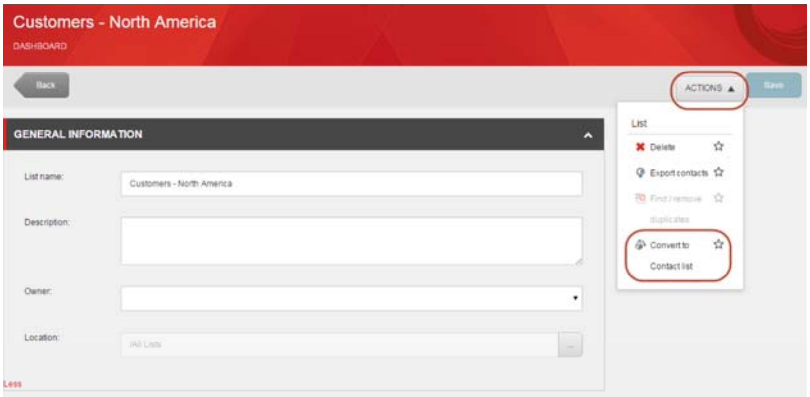
In the List Manager, you can convert a Segmented list to a Contact list, which means that you create a new Contact list that contains only the contacts that are in the Segmented list at the time you convert the list.

Important

When you convert a Segmented list to a Contact list, the Segmented list is *not* deleted. It will still be available to you in the list of Segmented lists.

To convert a Segmented list to a Contact list:

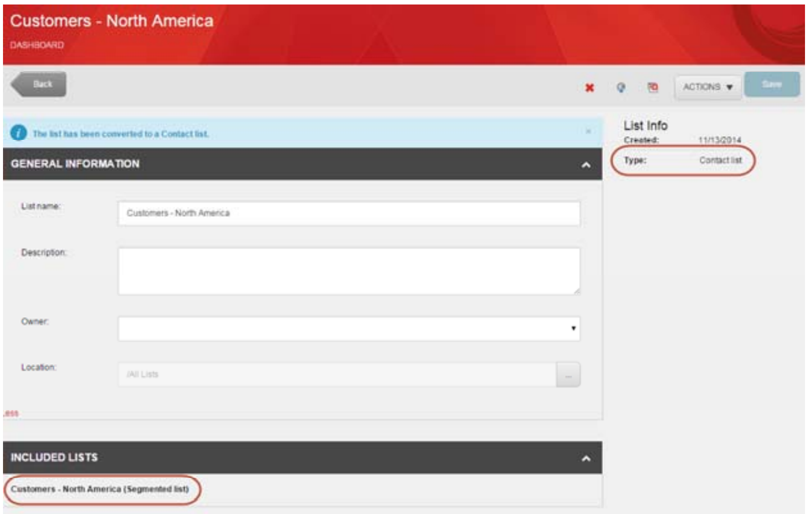
1. In the List Manager, navigate to and open the Segmented list that you want to convert.
2. On the list page, click Actions and then click Convert to Contact list.



Note

Depending on the number of contacts in the list, converting a Segmented list may take some time.

A notification appears to confirm that the Segmented list has been converted to a Contact list. Notice that the included list is the Segmented list and that in the List Info section, the *Type* has changed to *Contact list*.



3. To be able to distinguish the new Contact list from the Segmented list, enter a new name for the Contact list and save your changes.

   Note

   For converted Contact lists, you cannot include other list sources to the converted list or edit the current list source. However, you can include (or exclude) a converted Contact list as a list source to another list.

Send feedback about the documentation to docsite@sitecore.net.

Create a Contact list or a Segmented list

In the List Manager, you can create two types of lists – a Segmented list and a Contact list. For each list type, you can create two lists with different list sources included:

- Contact list
  - Contact list from existing list – create a new Contact list from an existing Contact list.
  - Empty Contact list – create a Contact list with no contacts included. For example, use the empty list to collect the visitors that sign up for a newsletter on your website or that trigger a message.
  - Create list from file – import contacts from a CSV file and create a Contact list that includes all the imported contacts.
- Segmented list
  - Segmented list from all contacts – create and segment a list from all the contacts in Sitecore.
  - Segmented list from existing list – create and segment a list from an existing Contact list.

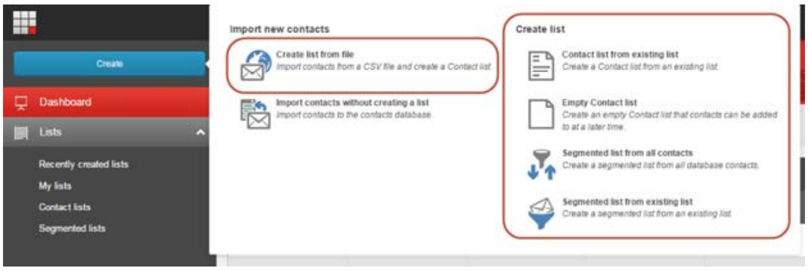
This topic outlines how to:

- Create a Contact list or a Segmented list
- Segment the included contacts in a Segmented list

Create a Contact list or a Segmented list

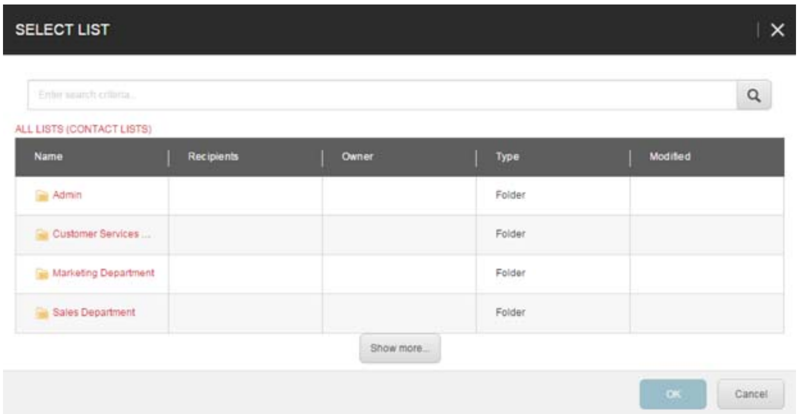
To create a list:

1. In the List Manager, click Create and in the menu that appears, in the Create list section, click the list type that you want to create.



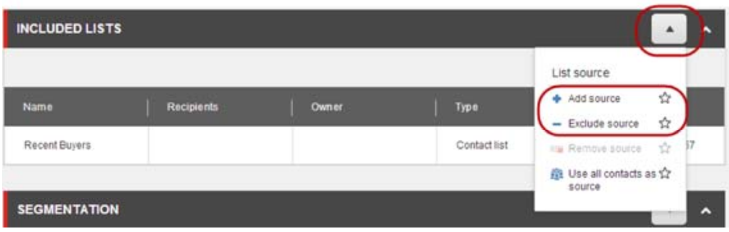
   - If you clicked Create list from file, import the new contacts in the Import Contacts wizard.
2. If you clicked Contact list from existing list or Segmented list from existing list, in the Select List dialog box, search for or navigate to the Contact list that you want to include in your new list and click OK.

   You can add or exclude additional lists when you have finished creating the list.



3. In the General Information section, enter a descriptive name for the list and click Save. If relevant, you can also add a description, owner, or change the location for the list.

4. For the new list, you can now add or exclude additional list sources. To add or exclude a Contact list to your new list, in the Included Lists section, click the drop-down arrow :
   - To add another Contact list, click Add source.
   - To exclude a Contact list, click Exclude list.



Note

List sources that are in the process of being exported, converted, or that are in use by another application cannot be added or excluded from your list.

5. In the Select list dialog box, search for or navigate to the relevant list and click OK. You can add or exclude as many list sources as you want but only one at a time.
6. Save your changes. After you have created a list with multiple sources, you should remove the duplicate contacts from the list to make sure that each contact only appear once in your list.

If you add new contacts to a list source of an existing contact list, the contact list is not updated accordingly. However, when you add or remove a list source of an existing contact list, the contact list is rebuilt, which means that all the new contacts in both existing and new list sources are added to the contact list. Depending on the number of contacts included in your list, this may take a considerable amount of time.

Note

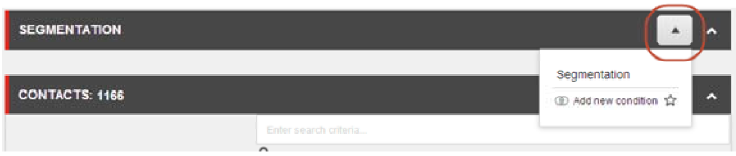
A Contact list is locked while it is built, rebuilt, or indexed and you can therefore not edit or use the list or view the contacts in the list until it is unlocked again.

Segment the included contacts in a Segmented list

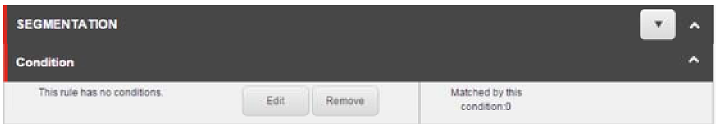
When you create a Segmented list, you must add conditions to the list to further segment the contacts in the included list source.

To add conditions to your Segmented list:

1. In the Segmentation section, click the drop-down arrow , and then click Add new condition.

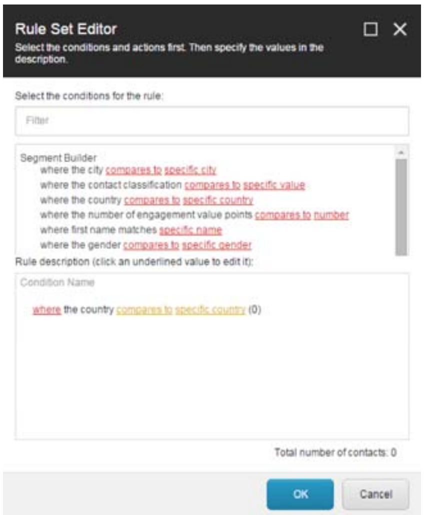


2. For the new condition, click Edit to set up the conditions to segment your included contacts by.



3. In the Rule Set Editor, in the Select the conditions for the rule field, select the relevant condition to segment your contacts by. For example, to segment your contacts by country, click the *where the country compares to specific country.*

   Alternatively, you can enter a search word to search for a specific condition.



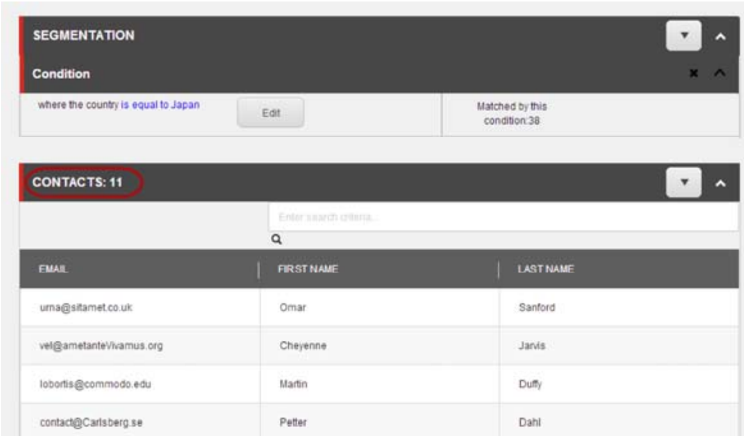
Note

You can add as many conditions in the rule description field as you want.

4. In the Rule description field, click the underlined values to specify the actions for the selected conditions.
5. When you have finished setting up the conditions and rules, click OK.

In the Segmentation section, you can see your conditions and rules that are used to segment the contacts in the Segmented list. The number Matched by this condition is the number of contacts in your entire database that meet your conditions.

6. Click Save. This updates the Contacts section to show only the contacts in the current contact list that meet the specified conditions.



Send feedback about the documentation to docsite@sitecore.net.

Find and remove duplicate contacts from a list

If you create a new Contact list by combining two or more lists, you can remove all the extra contracts with the same email address across all the included lists. In this way, you can make sure that no contacts receive the same email from you more than once.

Note

When you remove duplicate contacts from a list, the duplicates are only removed from the current list. All the contacts are still available in other lists and in the contacts database.

To find and remove duplicate contacts from a list:

1. In the List Manager, open the list that you want to remove duplicates from.
2. On the list page, click Actions and then click Find/remove duplicates.

A notification appears telling you how many duplicate contacts that were remove from the list.

3. Save your changes.

Send feedback about the documentation to docsite@sitecore.net.

Enable the export of custom facets to a CSV file

You can [create custom contact facets]() to include contact information that your organization wants to track, for example, addresses, country, or company.

If you want these custom contact facets to appear in the CSV file when you export lists of contacts from the List Manager, you must configure the `listManagement.exportContacts` pipeline.

Note

Make sure to [index new custom contact facets]() before you export them from the List Manager.

To enable the export of custom facets to a CSV file:

1. Create a new `GetCustomContactRows` class, making sure it is inherited from the existing class `Sitecore.ListManagement.ContentSearch.Pipelines.ExportContacts.GetContactRows, Sitecore.ListManagement.ContentSearch`.

   Override the protected `GetRow` method to export the relevant facets, for example, *Company*.

```
public class GetCustomContactRows : GetContactRows
 {
   protected override IEnumerable<string> GetRow(IEnumerable<ContactData> contacts)
   {
     Assert.ArgumentNotNull(contacts, "contacts");
     var sep = this.Delimiter.ToString(CultureInfo.InvariantCulture);
     yield return string.Join(sep, "Identifier", "FirstName", "Surname", "Email", "Company");
     foreach (var contact in contacts)
     {
       yield return string.Join(sep,
         contact.Identifier, contact.FirstName, contact.Surname,
         contact.PreferredEmail, contact["contact.Company"]);
     }
   }
 }
```

2. In the `App_Config\Include\ListManagement\Sitecore.ListManagement.config` file, in the `listManagement.exportContacts` pipeline, delete the existing `GetContactRows` processor and register the new one:

```
<listManagement.exportContacts>
  <processor    type="<custom namespace>.GetCustomContactRow, <custom assembly>" />
  <processor type="Sitecore.ListManagement.ContentSearch.Pipelines.ExportContacts.GetContactsStream, Sitecore.ListManagement.Cont
</listManagement.exportContacts>
```

Send feedback about the documentation to docsite@sitecore.net.

Import and export contacts from a list

In situations where you want to transfer contacts to Sitecore from a third party system (for example a CRM system) or where you have purchased a list of new leads from a marketing company, you can import these contacts from a CSV file directly into a list in the List Manager.

Similarly, if you want to transfer contacts from Sitecore to a third party system or create a backup of your contacts, you can export the contacts that are included in a list.

Note

For Sitecore to recognize special characters in the CSV file it should be encoded in UTF-8. Furthermore, Sitecore only supports comma as a separator.

This topic outlines how to:

- Import contacts
- Export contacts

Import contacts

In the List Manager, there are two ways to import contacts:

- Import contacts to the contacts database.
- Import contacts and add them to a new Contact list.

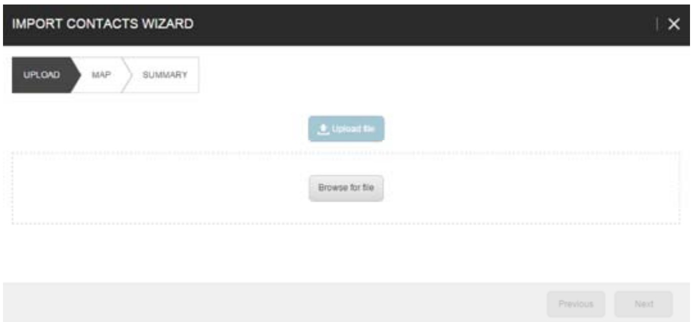
To import new contacts:

1. In the List Manager, click Create and in the menu that appears, in the Import new contacts section, click:
   - Import contacts to database to import contacts from a CSV file to your Analytics database.
   - Create list from file to import contacts from a CSV file and include the contacts in a new Contact list.

Note

The first row in a CSV file is a header and it is not imported.

2. In the Import Contacts wizard, on the Upload tab, click Browse for file.



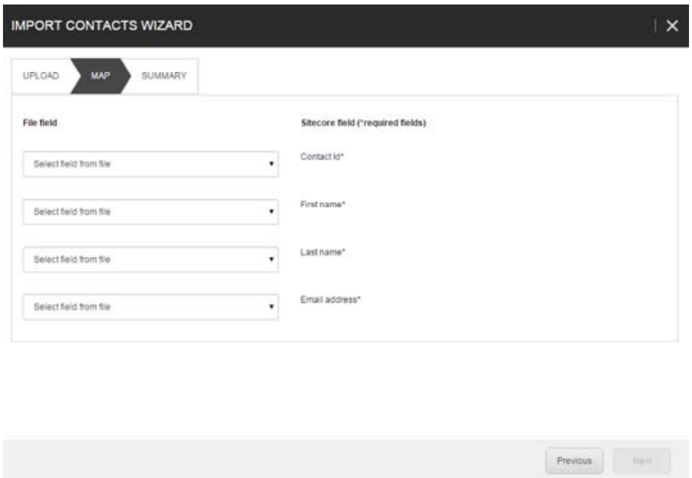
3. In the Open dialog box, click the CVS file with the new contacts and then click Open.

   Note

   You can import only one file at a time. If you want to upload a different file, you must click Remove to remove the file that is ready for upload.

4. Click Upload file and when the file has uploaded, click Next.
5. On the Map tab, map the fields in the file with the fields available in Sitecore.

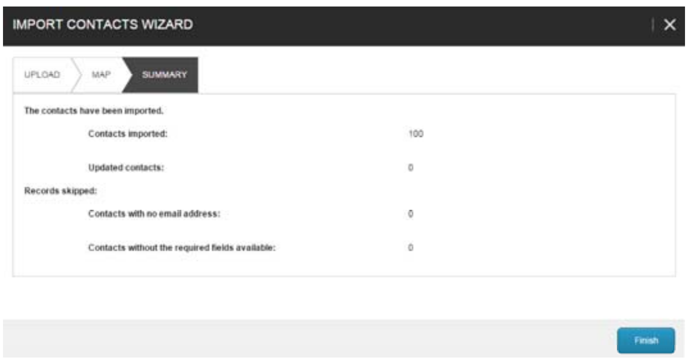
   For each required Sitecore field, in the File field drop-down menu, select the equivalent field in the CSV file. As the *Contact Id*, select the field in the import file that you want to use as the unique identifier for the contacts.



   Important

   List Manager uses the contact ID to identify the contacts in your database and to ensure that unnecessary duplicates are not created when you import new contacts. Therefore, before you import a list of contacts, it is important that you create a strategy for the contact ID. For example, if you use the contact's date of birth or zip code in combination with the first name, this makes a more suitable identifier than an email address alone. Using the email address as the contact ID can result in a contact appearing multiple times in the database if they have more than one email address.

6. When you have finished mapping the fields, click Next.
7. On the Summary tab, make sure that the import settings are correct and then click Finish.



   Note

   The *Updated contacts* shows the number of contacts in the import file that were already in the database and therefore only updated with any missing information.

   If you clicked Import contacts to database, the contacts are added to the database.

8. If you clicked Create list from file, the new contact list opens with the imported contacts included. Rename the new list and click Save.

Note

Depending on the number of contacts that you import, it may take some time to import and index all the contacts. You will get a system notification, when all the contacts have been included.

Export contacts

You can export contacts from either a Segmented list or a Contact list to a CSV file.

To export contacts from a list:

1. In the List Manager, find and open the list that contains the contacts that you want to export.
2. On the list page, click Actions and then click Export to a CSV file.

3. Find the downloaded CSV file in your default download folder on your local computer.

Send feedback about the documentation to docsite@sitecore.net.

Getting started with Sitecore IP Geolocation Service

If you are a marketer or a developer, and are interested in using the Sitecore IP Geolocation service with your Sitecore installation, then this article is intended for you.

With the Sitecore IP Geolocation services, visitors to your website benefit from a more enhanced, personalized experience with the many tools and activities that marketers can also benefit from.

With IP Geolocation you can set up personalization for different geographical regions and view reports pertinent to those locations. For visitors to your website, you can identify their:

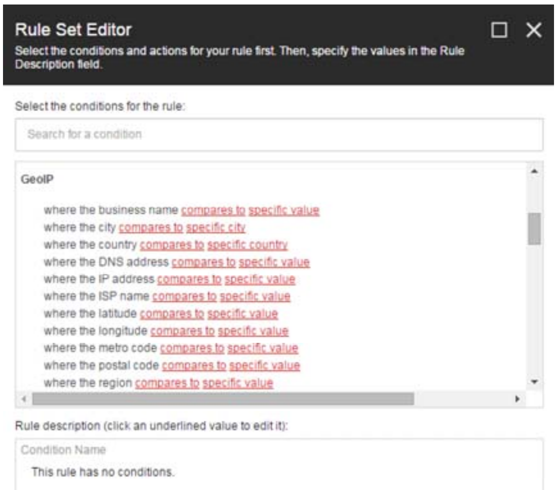
- Time zone
- Longitude and latitude
- Country
- Region
- Metro code
- City
- Post code
- Business name
- DNS address
- IP address
- ISP name

You can use Sitecore IP Geolocation to:

- Personalize site pages for visitors from different regions
- Fine tune tracking for Goals and Page events
- Define conditions in the automation of Engagement
- Geographically segment contact lists
- See geolocation reports
- Find geolocation details in contact experience profiles

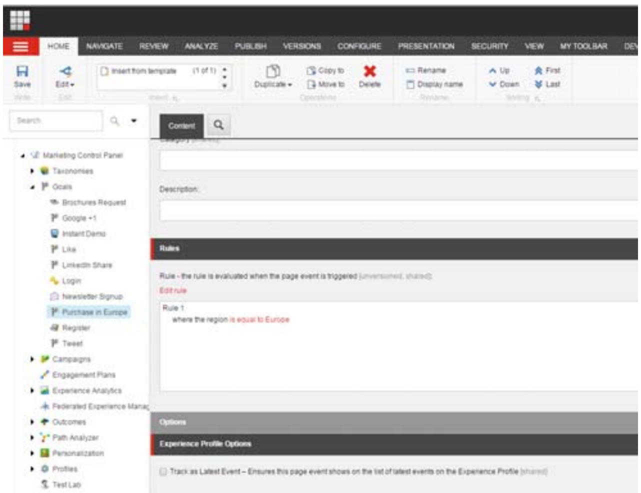
Personalize site pages for visitors from different regions

IP Geolocation is particularly useful for targeted marketing. You can set up personalization on pages by using the Rule Set Editor for IP Geolocation.



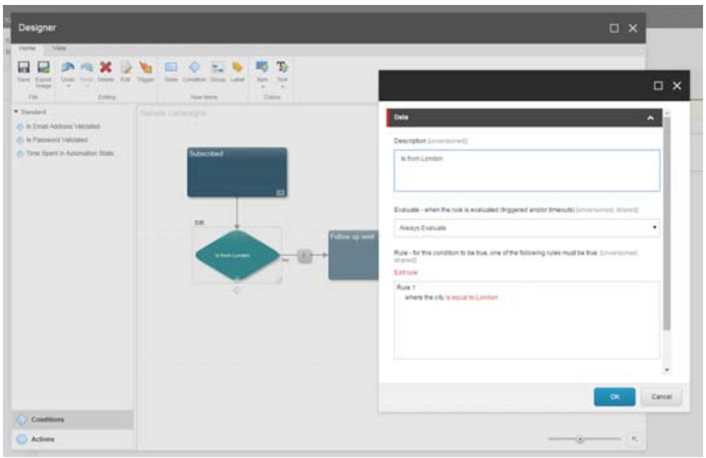
Fine tune tracking for Goals and Page Events

Another example of targeted marketing is that you can trigger goals and page events if a visitor from a certain location visits your website pages. You just apply the same rules that you use when creating a goal. Note that the goal in the screenshot below would only be triggered if the visitor came from Europe.



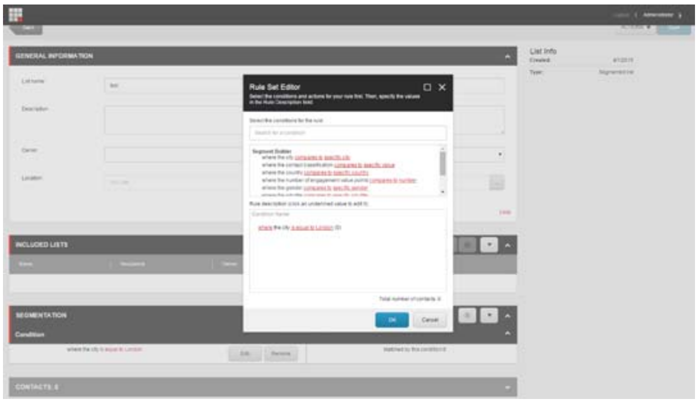
Define conditions in the automation of Engagement

A great tool for location-based campaigns, you can apply geolocation conditions to your engagement plans. For example, you could choose to send an email only to your contacts in or from London.



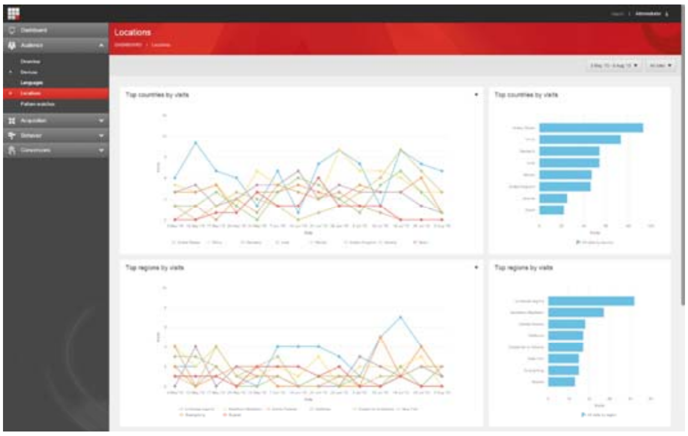
Geographically segment contact lists

You can also segment your lists by establishing rule sets, for example by location, gender, engagement value points, and so on.



See geolocation reports

Via the Experience Analytics tool, you can also generate reports from tracked visitor locations, giving you a better overview of the behavior of visitors at different locations.



Find geolocation details in contact experience profiles

Using the Sitecore IP Geolocation Service, you can gather geolocation details of the visitor such as the country, state, city, and the registered company name to build an Experience Profile for them.

Send feedback about the documentation to docsite@sitecore.net.

Setting up Sitecore IP Geolocation

The Sitecore IP Geolocation service provides information about the location and owner of an IP address beyond that provided by a reverse DNS lookup. IP Geolocation information includes the country, state, city, and the registered company name of every visitor.

You can use IP Geolocation lookups to create conditional renderings or personalization rules that show different content based on the visitor's location.

This topic describes:

- System requirements for Sitecore 8.0
- Activate the Sitecore IP Geolocation service
- Install the Sitecore IP Geolocation service for Sitecore 8.0
- Subscribe to the Sitecore IP Geolocation Service
- Install and enable the Sitecore IP Geolocation Service
- Upgrade your Sitecore IP Geolocation Service
- Configuration file changes
- Firewall configuration
- Using a proxy server

System requirements for Sitecore 8.0

The Sitecore IP Geolocation Service component is a separate package in Sitecore 8.0. It supports all Sitecore 8.0 versions from Sitecore 8.0 rev. 141212 and later. The module has been tested specifically with Sitecore 8.0 rev. 150223 (Update-2) and Sitecore 8.0 rev. 150427 (Update-3).

We recommend that you update your Sitecore installation to the latest version.

Activate the Sitecore IP Geolocation service

To activate the Sitecore IP Geolocation Service on Sitecore XP *, you must subscribe to the Sitecore IP Geolocation service in the App Center. The Sitecore IP Geolocation service is now free of charge and with unlimited lookups per month.

Note

Your IP Geolocation service subscription is connected to your unique Sitecore License ID. This means that if you update your license ID (for example, when you use a new license.xml), you must reactivate your IP Geolocation subscription.

Install the Sitecore IP Geolocation service for Sitecore 8.0

Installing Sitecore IP Geolocation Service consists of two basic steps:

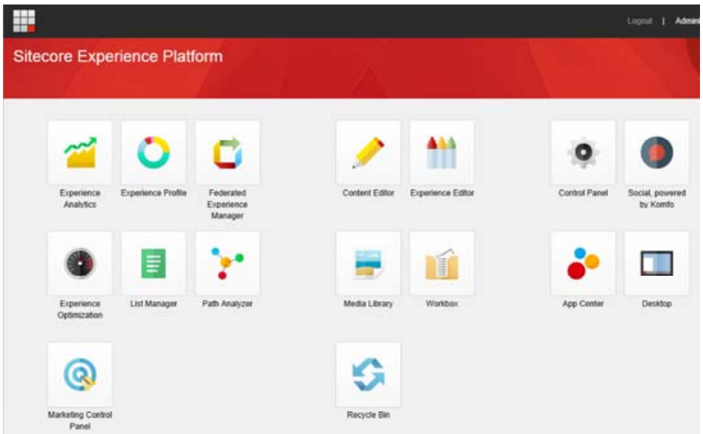
1. Subscribe to the Sitecore IP Geolocation Service
2. Install and enable the package. To ensure you have the right package, go to dev.sitecore.net.

Note

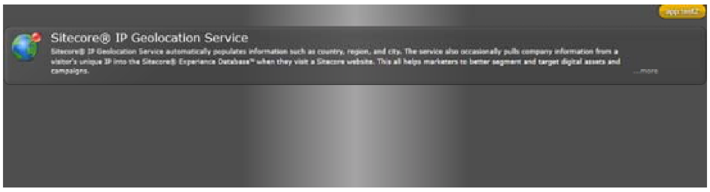
Your IP Geolocation service subscription is connected to your unique Sitecore License ID. This means that if you update your license ID (for example, when you use a new license.xml), you must reactivate your IP Geolocation subscription.

Subscribe to the Sitecore IP Geolocation Service

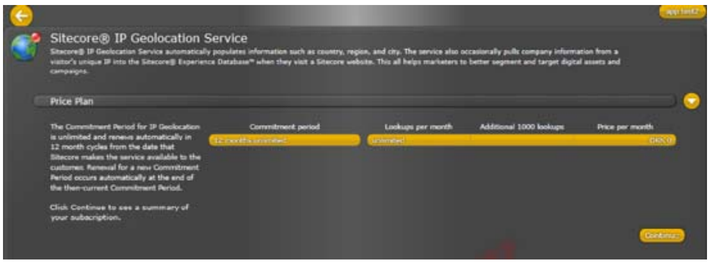
1. On the Sitecore Launchpad, click App Center.



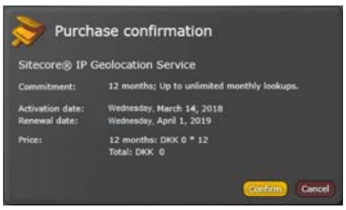
2. In the Sitecore App Center, click Sitecore IP Geolocation Service.



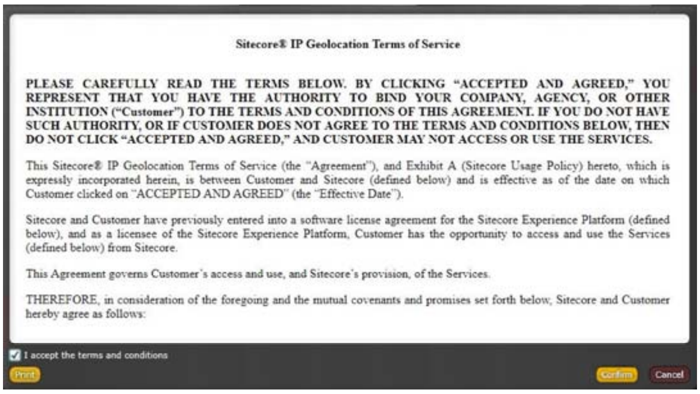
3. Click Continue to proceed with your sign-up and to see a summary of your subscription.



4. Click Confirm to view the Terms of Service.



5. Select the I accept the terms and conditions check box and click Continue to complete your subscription.


Sitecore® IP Geolocation Terms of Service

PLEASE CAREFULLY READ THE TERMS BELOW. BY CLICKING "ACCEPTED AND AGREED," YOU REPRESENT THAT YOU HAVE THE AUTHORITY TO BIND YOUR COMPANY, AGENCY, OR OTHER INSTITUTION ("Customer") TO THE TERMS AND CONDITIONS OF THIS AGREEMENT. IF YOU DO NOT HAVE SUCH AUTHORITY, OR IF CUSTOMER DOES NOT AGREE TO THE TERMS AND CONDITIONS BELOW, THEN DO NOT CLICK "ACCEPTED AND AGREED," AND CUSTOMER MAY NOT ACCESS OR USE THE SERVICES.

This Sitecore® IP Geolocation Terms of Service (the "Agreement"), and Exhibit A (Sitecore Usage Policy) hereto, which is expressly incorporated herein, is between Customer and Sitecore (defined below) and is effective as of the date on which Customer clicked on "ACCEPTED AND AGREED" (the "Effective Date").

Sitecore and Customer have previously entered into a software license agreement for the Sitecore Experience Platform (defined below), and as a licensee of the Sitecore Experience Platform, Customer has the opportunity to access and use the Services (defined below) from Sitecore.

This Agreement governs Customer's access and use, and Sitecore's provision, of the Services.

THEREFORE, in consideration of the foregoing and the mutual covenants and promises set forth below, Sitecore and Customer hereby agree as follows:

I accept the terms and conditions

Print

Confirm

Cancel

Now your subscription is complete.

Note

After activating your subscription, the system may take some time to update your license information. When this information is updated, the Sitecore IP Geolocation Service will automatically carry out geolocation lookups.

Install and enable the Sitecore IP Geolocation Service

After purchasing the Sitecore IP Geolocation Service from the Sitecore App Center, you can install the Sitecore IP Geolocation component on your Sitecore instance:

1. Go to `https://dev.sitecore.net/Downloads.aspx`, and download the *Sitecore IP Geolocation Service Client 1.2* package.
2. On the Sitecore Launchpad, click Control Panel, and in the Administration section, click Install a package and then use the wizard to install the package and accept the license.
3. Upload the package using the Upload Wizard.

   Important

   You must select the Overwrite all files checkbox.
4. Open the CES folder `/inetpub/wwwroot/yourinstancename/Website/App_Config/Include/CES/`

   Note

   Make sure you change `yourinstancename` to the actual name of your instance.
5. To enable the configuration files, remove `.disable` from the end of the file names.

   Enable the following files:

   - `Sitecore.CES.config`
   - `Sitecore.CES.GeoIp.config`
   - `Sitecore.CES.GeoIP.LegacyLocation.config`

Upgrade your Sitecore IP Geolocation Service

If you have the initial version of the Sitecore IP Geolocation client 1.2 rev. 150429 installed, you can use the following steps to upgrade to the Sitecore IP Geolocation Service Client 1.2 rev. 150602 (Update-1).

1. On `https://dev.sitecore.net/Downloads.aspx`, download `Sitecore IP Geolocation Service Client 1.2 rev. 150602`.
2. Follow steps 2–5 in the previous section.

Configuration file changes

If you have made any changes to your configuration files, you should:

- Back up your configuration files.
- Install the update package and overwrite the existing configuration files.
- Reapply your configuration file changes in the new version of your configuration files.

Firewall configuration

It is common to set up a firewall between your content management and content delivery servers. To ensure that the Sitecore IP Geolocation service works correctly in every scenario, you must configure your firewall settings to allow requests to the service.

Add a firewall rule to allow HTTPS protocol for:

- `geoIp-ces.cloud.sitecore.net`
- `Discovery-ces.cloud.sitecore.net`

Using a proxy server

To ensure that requests that come from a reverse proxy server to your Sitecore instance are tracked as the valid IP addresses of a client, and not as the IP addresses of a proxy:

- In `Sitecore.Analytics.Tracking.config` file, change the value of the `Analytics.ForwardedRequestHttpHeader` setting to `X-Forwarded-For`.

Send feedback about the documentation to docsite@sitecore.net.

Configure search results

You can configure search results in Sitecore to help users easily find items. You can specify how Sitecore displays search results and what happens when a user selects an item in a search result. This topic describes you can:

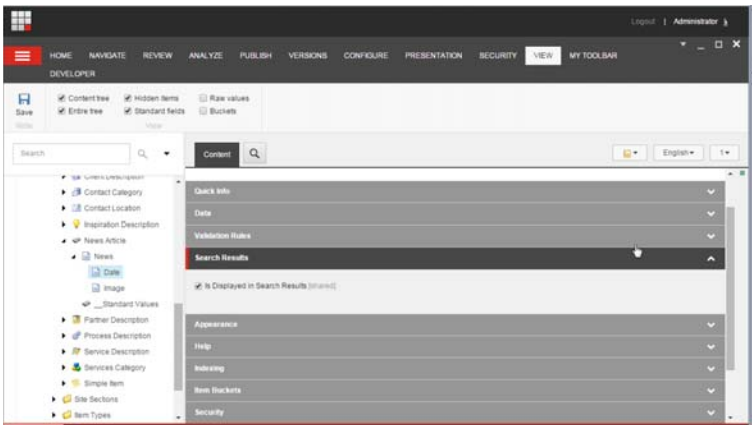
- Specify a field to display in search results
- Specify an image for search results
- Specify how Sitecore shows search results
- Specify how Sitecore opens an item in a search result
- Specify the number of items shown in a page
- Exclude the current item from search in item buckets

Specify a field to display in search results

You can specify which fields Sitecore displays in search results.

To specify that a field should be displayed in the search results:

1. Open the template for the item.
2. On the View tab, in the View group, select the Standard Fields check box.
3. In the content tree, expand the template and select the field that you want to display in the search results.
4. On the Content tab, scroll down to the Search Results section:



5. In the Search Results section, select the Is Displayed in Search Results check box.

Note

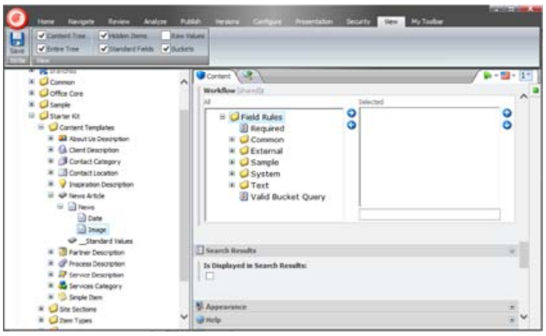
These values are cached. Therefore, you must clear the cache after you have made changes to this setting by resetting the cache in `/sitecore/admin/cache.aspx`.

Specify an image for search results

Sitecore displays an image with each content item listed in a search result. By default, Sitecore uses the icon of the template of the item. You can specify that Sitecore shows another image field from a template in search results.

To specify which image field Sitecore displays in search results:

1. Open the template for the item.
2. On the View tab, in the View group, select the Standard Fields check box.
3. In the content tree, navigate to the template and expand it.
4. Select the image field that you want to appear in the search results.
5. On the Content tab, scroll down to the Search Results section.



6. In the Search Results section, select the Is Displayed in Search Results check box.

The image that Sitecore displays in this field is shown in the search results.

Note

These values are cached. Therefore, you must clear the cache after you have made changes to this setting by resetting the cache in `/sitecore/admin/cache.aspx`.

Specify how Sitecore shows search results

Sitecore comes with different views that you can use to display search results.

The three default views are:

- List – The default view. Sitecore shows items as a plain list.
- Grid – Sitecore shows item information in rectangular cells that are laid out in rows, and each row contains several columns of cells.
- Image – Sitecore shows item images in a grid. If an item is an image, this image is shown. Other items are represented by their icons.

You can also specify the following views, but you must enable them first:

- ID – The ID view is useful for developers and administrators. You can use it as a quick way of assigning multiple values to a list. When you click the ID view for a search, you get a list of item IDs that you can copy and paste into a multilist field, for example.
- Lock – Similar to the Grid view but also shows whether the items in the result set are locked or not.
- Media – Similar to the image view, but for media such as videos.
- Preview – Show a preview of each item in the result (what the item looks like in context on the website.)
- Table – Show the items in the result as a table.
- Tag – This view orders items in the result by tags.

To enable a view:

1. In the Content Editor, navigate to `/sitecore/system/settings/Buckets/Views` and select the view that you want to enable.
2. In the View Details section, select the Enabled check box.
3. If you want this to be the default view, select the Default check box. You must enable the default view.

To select views for an item:

1. Select the item in the Content Editor.
2. On the Content tab, scroll down to the Item Buckets section.
3. In the Enabled views field, select the views you want to enable.
4. Select a view in the Default view field.

Specify how Sitecore opens an item in a search result

When users click an item in a search result, the default behavior is that Sitecore opens the item in a new tab in the Content Editor and makes this the selected tab.

To specify how Sitecore opens items from search results:

1. Open the `/sitecore/System/Settings/Buckets/Item Buckets Settings` item.
2. In the Show Search Results In field, select:
   - *New Tab* – Sitecore opens the item in a new tab and makes this the selected tab. This is the default behavior.
   - *New Content Editor* – Sitecore opens the item in a new instance of the Content Editor.
   - *New Tab Not Selected* – Sitecore opens the item in a new tab but it is not made the selected tab.

Specify the number of items shown in a page

Sitecore displays search results with 20 items per page by default.

To specify the number of items shown per page in search results:

1. Open the `Sitecore.Buckets.config` file in the `Website\App_Config\Include` folder.
2. Change the number in the following setting:

```
<setting name="BucketConfiguration.DefaultNumberOfResultsPerPage" value="20"/>
```

Exclude the current item from search in item buckets

When you search within an item bucket, by default the item you search from (the item that is selected when the search starts) is included is the search results when it matches the search.

To exclude the current item from the search results:

- In the `Sitecore.Buckets.config` file, set the `BucketConfiguration.ExcludeContextItemFromResult` setting to true.

Send feedback about the documentation to docsite@sitecore.net.

Configuring search and item bucket scalability

This topic explains the settings related to search and item bucket scalability in the following configuration files:

- `Sitecore.Buckets.config`
- `Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config`
- `Web.config`

Sitecore.Buckets.config

The `Sitecore.Buckets.config` file contains the following settings which you can use to improve the scalability of your Sitecore solution:

Setting	Description	Example
`BucketTriggerCount`	If you enable the `AutoBucket` events, this setting specifies the maximum number of children that an item can have before it is automatically converted into an item bucket.	`<setting name="BucketConfiguration.BucketTriggerCount" value="100"/>` This setting means that when an item has 100 children, Sitecore asks if you want to automatically convert this item into an item bucket.
`BucketTemplateId`	Specifies the template of a folder item that contains all the hidden bucketable items as the GUID of the new folder item. We recommend you use the default value.	`<setting name="BucketConfiguration.BucketTemplateId" value="{ADB6CA4F-03EF-4F47-B9AC-9CE2BA53FF97}" />`
`SecuredItems`	This setting determines what happens to search results that are returned when a user does not have permission to access them. The available options are "hide" and "blur".	`<setting name="BucketConfiguration.SecuredItems" value="hide"/>`

Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config

The `Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config` file contains the following setting that you can use to improve the scalability of your Sitecore solution:

Setting	Description	Example
`ContentSearch.LuceneQueryClauseCount`	This setting specifies the clause count. Increasing this value increases memory consumption. Only increase this value if you need to run very large queries.	`<setting name="ContentSearch.LuceneQueryClauseCount" value="1024"/>`

web.config

The `web.config` file contains the following setting that you can use to improve the scalability of your Sitecore solution:

Setting	Description	Example
`Indexing.UpdateInterval`	Specifies how often the Web database index is updated. Set an index update interval to record when unstructured items are created, deleted or modified in the Web database. If you do not set an update interval these events are not automatically included in the index.	`<setting name="Indexing.UpdateInterval" value="00:05:00"/>`

Send feedback about the documentation to docsite@sitecore.net.

Create a default and a persistent search query

You can specify a default search that Sitecore runs when users open a new search tab. You can also prevent users from being able to remove parts of a query by making the search query *persistent*.

Note

You can define both default and persistent queries for an item.

This topic outlines how to:

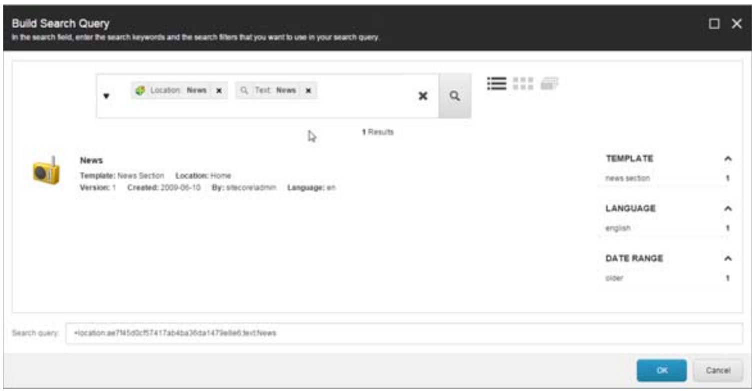
- Define a default search
- Make a search query persistent

Define a default search

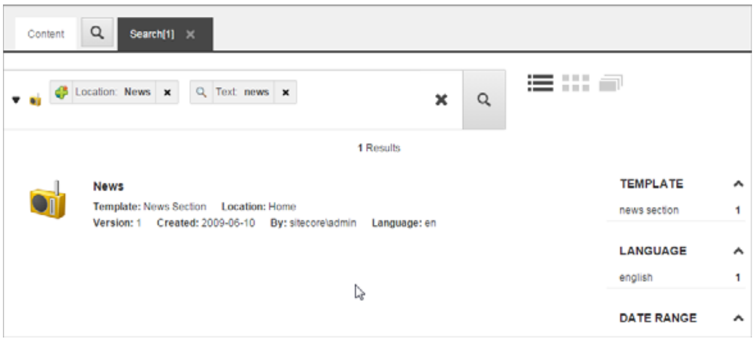
To define a default search:

1. In the content tree, navigate to the item you want to create a default search for.
2. On the Content tab, in the Item Buckets section, in the Default bucket query field, enter a query. For example:

You can enter a query manually, paste in a query, or you can build a query in the Build Search Query dialog:



When users open a new search tab, it looks, for example, like this:



Make a search query persistent

When users open a new search tab, they can delete the filters in the query by clicking X. If you do not want users to be able to remove parts of the query, you can make the search query *persistent*.

To make a search query persistent:

1. In the content tree, navigate to the item you want to define a persistent query for.
2. On the Content tab, in the Item Buckets section, in the Persistent bucket filter field, enter a query.

Sitecore supports these filters:

- tag
- template
- location
- sort
- custom
- tag
- start
- end

You can add multiple filters with a semicolon between the filters.

Note

Every filter is case sensitive.

For example, to search for the keyword `pineapple` between a start date of `03/03/2012` and an end date of `04/04/2012`, the filter string is:

```
text:pineapple;start=03/03/2012;end=04/04/2012
```

Send feedback about the documentation to docsite@sitecore.net.

Search result boosting

You can boost specific items and fields when they are indexed at indexing time. These items and fields will score better than other items and fields. You can use boosting, for example, to promote the most popular or the most sold books in a bookshop in a search result.

You can apply boosting at an item level or a field level.

You use field-level boosting to specify that matches on some fields are more or less important than matches on other fields. For example, for a bookshop, the book product item has three fields: Title, Summary and Foreword. When searching, matches in the Foreword field should be less important than matches in the Title and Summary fields.

Note

You can also apply boosting at query time. This boosting also takes effect for items or fields that you have boosted at indexing time.

Configuration

You configure boosting in the `Sitecore.ContentSearch.config` in the `Website\App_Config\Include` folder. When you configure boosting, note the following:

- Default value for Boost is 1
- Values greater than 1 lift results to the top.
- Values less than 1 move results to the bottom.

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
  <sitecore>
    ...
    <pipelines>
      <!-- RESOLVE FIELD LEVEL BOOSTING
           Pipeline for resolving boosting rules on fields.
           Arguments: (Item) Item being indexed
           Example : Boost search results by a field value.
      -->
      <indexing.resolveFieldBoost help="Processors should derive from
                Sitecore.ContentSearch.Pipelines.ResolveBoost.ResolveFieldBoost.
                BaseResolveFieldBoostPipelineProcessor">
         <processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost
                .ResolveFieldBoost.SystemFieldFilter, Sitecore.ContentSearch"/>
         <processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
                 ResolveFieldBoost.FieldDefinitionItemResolver, Sitecore.ContentSearch"/>
         <processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
                 ResolveFieldBoost.StaticFieldBoostResolver, Sitecore.ContentSearch"/>
      </indexing.resolveFieldBoost>
      <!-- RESOLVE ITEM LEVEL BOOSTING
           Pipeline for resolving boosting rules on items.
           Arguments: (Item) Item being indexed
           Example : Boost search results by an Item Template.
      -->
      <indexing.resolveItemBoost help="Processors should derive from
              Sitecore.ContentSearch.Pipelines.ResolveBoost.
              ResolveItemBoost.BaseResolveItemBoostPipelineProcessor">
        <processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
                ResolveItemBoost.ItemLocationFilter, Sitecore.ContentSearch">
           <includedLocations hint="list">
              <content>/sitecore/content</content>
              <media>/sitecore/media library</media>
           </includedLocations>
        </processor>
        <processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
                ResolveItemBoost.StaticItemBoostResolver, Sitecore.ContentSearch"/>
        <processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
                ResolveItemBoost.LocalRuleBasedItemBoostResolver, Sitecore.ContentSearch"/>
        <processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
```

```
                ResolveItemBoost.GlobalRuleBasedItemBoostResolver, Sitecore.ContentSearch"/>
      </indexing.resolveItemBoost>
    </pipelines>
    <!-- BOOSTING MANAGER
         The manager class controlling the boosting resolution logic
    -->
    <boostingManager defaultProvider="default" enabled="true">
      <providers>
        <clear/>
        <add name="default" type="Sitecore.ContentSearch.Boosting.
                PipelineBasedBoostingProvider, Sitecore.ContentSearch"/>
      </providers>
    </boostingManager>
    ...
  </sitecore>
</configuration>
```

Note

Do not make changes directly to the configuration files. Instead, you must create a patch file that performs the required changes during runtime.

Field-level boosting

The `indexing.resolveFieldBoost` pipeline resolves field-level boosting. The following processors are enabled in this pipeline:

```
<processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
   ResolveFieldBoost.SystemFieldFilter, Sitecore.ContentSearch"/>
<processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
  ResolveFieldBoost.FieldDefinitionItemResolver, Sitecore.ContentSearch"/>
<processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
  ResolveFieldBoost.StaticFieldBoostResolver, Sitecore.ContentSearch"/>
```

The `SystemFieldFilter` processor ensures that all the system fields are ignored (fields with names that start with the “__” characters).

The `FieldDefinitionItemResolver` processor resolves the field-definition item that provides the boost value.

Once the template definition item has been resolved, the `StaticFieldBoostResolver` is started. This processor reads the boost value from the field-definition item.

To see the Indexing section of items, you must select the Standard Fields check box on the View tab in the Content Editor.

Item-level boosting

The `indexing.resolveItemBoost` pipeline resolves item-level boosting. The following processors are enabled in this pipeline:

```
<processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
      ResolveItemBoost.ItemLocationFilter, Sitecore.ContentSearch">
    <includedLocations hint="list">
      <content>/sitecore/content</content>
      <media>/sitecore/media library</media>
    </includedLocations>
</processor>
<processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
      ResolveItemBoost.StaticItemBoostResolver, Sitecore.ContentSearch"/>
<processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
      ResolveItemBoost.LocalRuleBasedItemBoostResolver, Sitecore.ContentSearch"/>
<processor type="Sitecore.ContentSearch.Pipelines.ResolveBoost.
      ResolveItemBoost.GlobalRuleBasedItemBoostResolver, Sitecore.ContentSearch"/>
```

The `ItemLocationFilter` processor ensures that only items in the specified locations are processed by the pipeline. You can extend this list in this way:

```
<includedLocations hint="list">
   <content>/sitecore/content</content>
   <media>/sitecore/media library</media>
   <custom>/sitecore/custom</custom>
</includedLocations>
```

The `StaticItemBoostResolver` processor reads the boost value of the indexed item.

This value can also be set for all items based on a template using __Standard Values fields. To see the Indexing section of items, you must select the Standard Fields check box on the View tab in the Content Editor. The Boost Value field is in this section.

Rule-based boosting

The `LocalRuleBasedItemBoostResolver` and the `GlobalRuleBasedItemBoostResolver` processors provide rule-based boosting for indexed items. These two processors are both executed using the rules engine. They look up boosting rules in different locations:

- The `LocalRuleBasedItemBoostResolver` processor reads rules from the Boosting rules field in the Indexing section.

  The source of the Boosting rules field is restricted to the following location where all boosting rules are managed:

  `/sitecore/system/Settings/Rules/Indexing and Search/Item Rules`

  This set of boosting rules can also be set on the __Standard Values level, just as other field values.

- The `GlobalRuleBasedItemBoostResolver` processor processes all boosting rules created under `/sitecore/system/Settings/Indexing and Search/Rules/Global Rules`.

For performance reasons, use the local boosting rules as much as possible.

Create a boosting rule

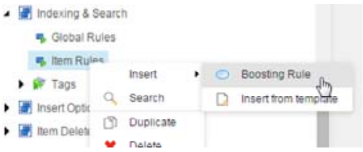
You can create a boosting rule that is used locally per item, or a global rule. You create global boosting rules as children of the Global Rules item; you create local boosting rules as children of the Item Rules item.

To create a boosting rule:

1. Navigate to this item in the content tree:

   `/sitecore/system/Settings/Rules/Indexing and Search/`

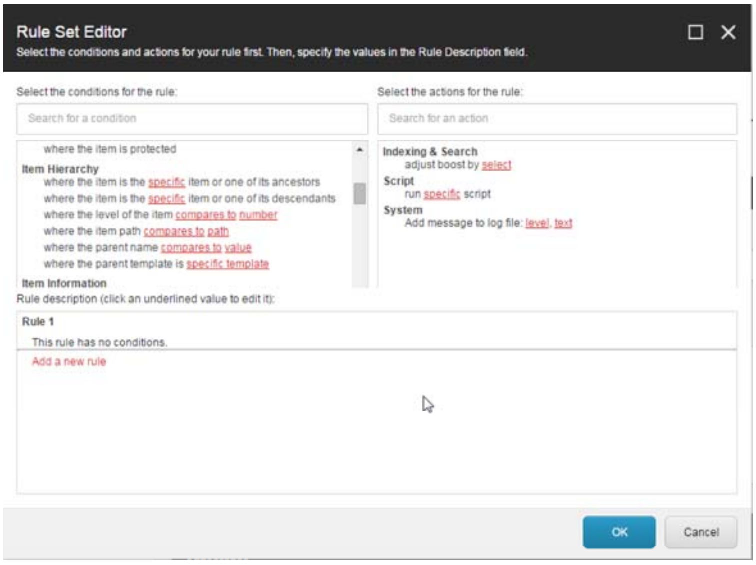
2. To create a new local item rule, right-click on the Item Rules item and then click Insert, Boosting Rule:

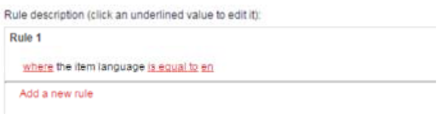
3. In the Message dialog, enter a name for the new rule. For example, you can create a boosting rule that boosts all items in English:

4. Select the new rule and in the Data section, click Edit rule. The Rule Set Editor opens:

5. Select the conditions for your rule. For example, to create a rule that boosts all items in English, in the Item Version section, select the *where the item language compares to value* condition from the list and then edit the condition as follows:

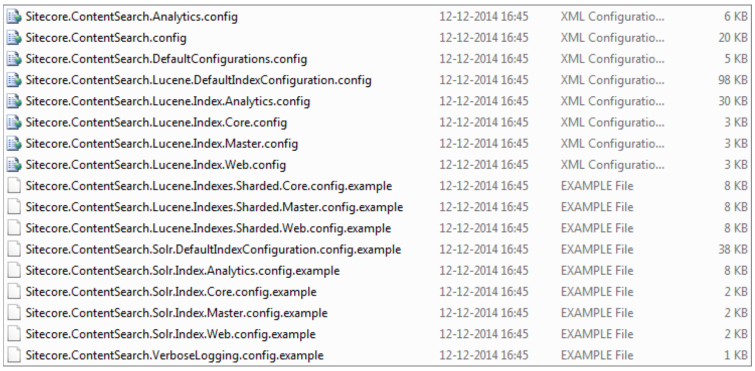
6. In the Indexing & Search section, select the *adjust boost by select* action:

7. In the Rule description section, click *select* and adjust the boost value. The boost value can be either a positive or a negative number.

Send feedback about the documentation to docsite@sitecore.net.

The search and indexing configuration files

This topic describes the configuration files that are related to search and indexing. The following files are shipped with Sitecore CMS 8.0. They are in the `Website\App_Config\Include` folder.



Sitecore.ContentSearch.Analytics.config	12-12-2014 16:45	XML Configuratio...	6 KB
Sitecore.ContentSearch.config	12-12-2014 16:45	XML Configuratio...	20 KB
Sitecore.ContentSearch.DefaultConfigurations.config	12-12-2014 16:45	XML Configuratio...	5 KB
Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config	12-12-2014 16:45	XML Configuratio...	98 KB
Sitecore.ContentSearch.Lucene.Index.Analytics.config	12-12-2014 16:45	XML Configuratio...	30 KB
Sitecore.ContentSearch.Lucene.Index.Core.config	12-12-2014 16:45	XML Configuratio...	3 KB
Sitecore.ContentSearch.Lucene.Index.Master.config	12-12-2014 16:45	XML Configuratio...	3 KB
Sitecore.ContentSearch.Lucene.Index.Web.config	12-12-2014 16:45	XML Configuratio...	3 KB
Sitecore.ContentSearch.Lucene.Indexes.Sharded.Core.config.example	12-12-2014 16:45	EXAMPLE File	8 KB
Sitecore.ContentSearch.Lucene.Indexes.Sharded.Master.config.example	12-12-2014 16:45	EXAMPLE File	8 KB
Sitecore.ContentSearch.Lucene.Indexes.Sharded.Web.config.example	12-12-2014 16:45	EXAMPLE File	8 KB
Sitecore.ContentSearch.Solr.DefaultIndexConfiguration.config.example	12-12-2014 16:45	EXAMPLE File	38 KB
Sitecore.ContentSearch.Solr.Index.Analytics.config.example	12-12-2014 16:45	EXAMPLE File	8 KB
Sitecore.ContentSearch.Solr.Index.Core.config.example	12-12-2014 16:45	EXAMPLE File	2 KB
Sitecore.ContentSearch.Solr.Index.Master.config.example	12-12-2014 16:45	EXAMPLE File	2 KB
Sitecore.ContentSearch.Solr.Index.Web.config.example	12-12-2014 16:45	EXAMPLE File	2 KB
Sitecore.ContentSearch.VerboseLogging.config.example	12-12-2014 16:45	EXAMPLE File	1 KB

Sitecore.ContentSearch configuration file

The `Sitecore.ContentSearch.config` file has the following elements that you can specify:

Index element	Description
`<events>`	Specifies event handlers for `indexing:start`, `indexing:start:remote`, `indexing:end`, and `indexing:end:remote` events.
`<hooks>`	Runs necessary initialization processes, such as EventHub registration and warmup queries for you index. You can extend this class or add additional hooks.
`<pipelines>`	Specifies all pipeline processors related to search and indexing.
`<boostingManager>`	Specifies the manager class controlling the search result boosting logic.
`<searchManager>`	Specifies the search manager and provider classes.
`<scheduling>`	Specifies the interval at which indexing will occur.
`<settings>`	Specifies settings, such as parallel indexing, maximum search results, and date format.
`<commands>`	Specifies handlers for indexing events.

Default index configuration file

The default index configuration file has the following elements that you can specify:

Index element	Description
`<indexUpdateStrategies>`	Specifies which update strategy will be used.
`<databasePropertyStore>`	Specifies key values that can be used in a search.
`<configuration>`	Settings for index configuration.

<DefaultIndexConfiguration>	Settings used if no custom settings are available for an index.
<IndexAllFields>	Default value is true.
<Analyzer>	The Lucene Standard Analyzer is specified by default. A custom analyzer might speed up searches.
<fieldMap>	Enables you to map a Sitecore field name to the index and store it in an appropriate way, including original value storage, boost value, and data type.
<virtualFieldProcessors>	Specifies custom query processing for a named field.
<exclude>	Excludes items from the index based on template type. Excludes specific fields from the index, if `IndexAllFields` is set to true.
<include>	Includes items from the index based on template type, if `IndexAllFields` is set to false.
<fields>	Enables you to format the way that field values are stored in the index, remove inbuilt Sitecore fields, and store computed fields.

Send feedback about the documentation to docsite@sitecore.net.

Index update strategies

You use index update strategies to maintain indexes. You can configure each index with a unique set of index update strategies. You should not specify more than three update strategies per index for performance reasons.

Sitecore provides a varied set of index update strategies, and you can extend this set with more strategies. All the strategies that are delivered with Sitecore are defined under the following node in the `Sitecore.ContentSearch` configuration files:

```
sitecore/contentSearch/indexConfigurations/indexUpdateStrategies
```

```
<manual type="Sitecore.ContentSearch.Maintenance.Strategies.ManualStrategy,           Sitecore.ContentSearch" />
```

Sitecore comes with the following strategies:

- RebuildAfterFullPublish
- OnPublishEndAsync
- IntervalAsynchronous
- Synchronous
- RemoteRebuild
- TimedIndexRefresh
- Manual

RebuildAfterFullPublish strategy

This strategy is defined in the following way in the configuration file:

```
<rebuildAfterFullPublish type="Sitecore.ContentSearch.Maintenance.Strategies. RebuildAfterFullPublishStrategy,
Sitecore.ContentSearch" />
```

During initialization, this strategy subscribes to the `OnFullPublishEnd` event and it triggers a full index rebuild.

In a distributed environment, the index rebuild is triggered on all remote servers where this strategy is configured. In this case, you must enable the event queue.

In environments where a full publish is required to run regularly, you should not trigger incremental index rebuilds because this uses a lot of resources. Instead, this strategy triggers a full index rebuild when a full publish process has completed.

When you attach this strategy to an index, you will see the following message in the `CrawlingLog` file when it is initialized:

```
Initializing RebuildAfterFullPublishStrategy for index '<index_name>'
```

When this strategy is triggered, you will see the following message in the `CrawlingLog` file:

```
RebuildAfterFullPublishStrategy triggered on index '<index_name>'
```

Attaching the RebuildAfterFullPublish strategy to an index

Attach this strategy to an index in the following way:

```
<index id="sitecore_index" type="Sitecore.ContentSearch.LuceneProvider.
          LuceneIndex, Sitecore.ContentSearch.LuceneProvider">
   <param desc="name">$(id)</param>
   <param desc="folder">$(id)</param>
   <strategies hint="list:AddStrategy">
```

```
      <strategy ref="contentSearch/indexConfigurations/indexUpdateStrategies/rebuildAfterFullPublish" />
   </strategies>
   <Analyzer ref="search/analyzer" />
```

Best practice

You should not combine this strategy with the Synchronous Strategy, but you can combine it with any of the other strategies.

Because this strategy causes a full index rebuild, you should combine it with the SwitchOnRebuildLuceneIndex or the SwitchOnRebuildSolrSearchIndex.

When you use this strategy in combination with the OnPublishEndAsync strategy, you need to register it as the first entry in the list for it to be triggered first:

```
<index id="sitecore_index" type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex,
          Sitecore.ContentSearch.LuceneProvider">
   <param desc="name">$(id)</param>
   <param desc="folder">$(id)</param>
   <strategies hint="list:AddStrategy">
      <strategy ref="contentSearch/indexConfigurations/indexUpdateStrategies/rebuildAfterFullPublish" />
      <strategy ref="contentSearch/indexConfigurations/indexUpdateStrategies/onPublishEndAsync" />
   </strategies>
   <Analyzer ref="search/analyzer" />
```

OnPublishEndAsync strategy

This strategy is defined in the following way in the configuration file:

```
<onPublishEndAsync type="Sitecore.ContentSearch.Maintenance.Strategies.
           OnPublishEndAsynchronousStrategy, Sitecore.ContentSearch">
          <param desc="database">web</param>
          <CheckForThreshold>true</CheckForThreshold>
</onPublishEndAsync>
```

During initialization, this strategy subscribes to the `OnPublishEnd` event and triggers an incremental index rebuild.

If you have separate CM and CD servers, this event is triggered via the `EventQueue` object. This means that you must enable the `EventQueue` object for this strategy to work in this kind of environment.

Note

There is an additional `database` parameter that is passed to the constructor of the `OnPublishEndAsynchronousStrategy` class. This parameter defines the database to look up the item changes from.

When you attach this strategy to an index and it is initialized, you will see the following message in the `CrawlingLog` file:

```
Initializing OnPublishEndAsynchronousStrategy for index '<index_name>'.
```

When this strategy is triggered, you will see the following message in the `CrawlingLog` file:

```
"<index_name> OnPublishEndAsynchronousStrategy executing."
```

Processing

This strategy uses the `EventQueue` object from the database it was initialized with:

```
<param desc="database">web</param>
```

This means that this strategy depends on a number of things:

- This database must be specified in the `<databases />` section of the configuration file.
- The `EnableEventQueues` setting must be `true`.
- The `EventQueue` table within the preconfigured database must have entries that are dated later than the last update timestamp of the index.

The strategy forces a full index rebuild when the number of entries in the history table exceeds the number defined in the `Indexing.FullRebuildItemCountThreshold` setting. This is to prevent excessive processing of the event queue. In most cases, this happens because there was a large publishing or deployment, and this should always trigger a full index rebuild.

This behavior is only triggered when the following property in the configuration file is set to `true` (and this is the default):

```
<CheckForThreshold>true</CheckForThreshold>
```

If you specify this setting as `true`, we recommend that you also use the SwitchOnRebuildLuceneIndex or the SwitchOnRebuildSolrSearchIndex implementation for any index that uses this strategy.

The value of the `Indexing.FullRebuildItemCountThreshold` setting has a default of 100000.

Attaching the OnPublishEndAsync strategy to an index

Attach this strategy to an index in the following way:

```
<index id="sitecore_index" type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex,
         Sitecore.ContentSearch.LuceneProvider">
   <param desc="name">$(id)</param>
```

```
   <param desc="folder">$(id)</param>
   <strategies hint="list:AddStrategy">
      <strategy ref="contentSearch/indexConfigurations/indexUpdateStrategies/onPublishEndAsync" />
   </strategies>
   <Analyzer ref="search/analyzer" />
```

Best practice

Do not combine this strategy with any of these strategies:

- Synchronous
- IntervalAsynchronous

You can combine it with these strategies:

- RebuildAfterFullPublish
- RemoteRebuild

You should use this strategy for multiserver/multi-instance environments, where you have already enabled the *EventQueue*.

IntervalAsynchronous strategy

This strategy is defined in the following way in the configuration file:

```
<intervalAsyncMaster type="Sitecore.ContentSearch.Maintenance.Strategies.
          IntervalAsynchronousStrategy, Sitecore.ContentSearch">
   <param desc="database">master</param>
   <param desc="interval">00:00:10</param>
   <CheckForThreshold>true</CheckForThreshold>
</intervalAsyncMaster>
```

- You specify the database to look up item changes for the processing from with the `database` parameter.
- You specify the frequency of the strategy trigger with the `interval` parameter.

When you attach this strategy to an index and it is initialized, you can see the following message in the `CrawlingLog` file:

```
Initializing IntervalAsynchronousUpdateStrategy for index '<index_name>'.
```

When this strategy is triggered, you can see the following message in the `CrawlingLog` file:

```
IntervalAsynchronousUpdateStrategy triggered on index '<index_name>'
```

Processing

This strategy is triggered by a time interval and not the `OnPublishEnd` event. It uses the `EventQueue` table of the source database. The `source` database is specified by the `database` parameter of the strategy. For example:

```
<param desc="database">web</param>
```

The preconditions for using this strategy are:

- The `EnableEventQueues` setting must be `true`.
- The referenced database must be defined in the `<databases>` configuration section
- The referenced database must match at least one database that is defined in a search index to be crawled

The strategy uses an internal timer that is initialized with a predefined interval value. The strategy is triggered when the timer fires. In this example, the timer is set to fire every 10 seconds:

```
<intervalAsync type="Sitecore.ContentSearch.Maintenance.Strategies.
         IntervalAsynchronousStrategy, Sitecore.ContentSearch">
   <param desc="database">web</param>
   <param desc="interval">00:00:10</param>
   <CheckForThreshold>true</CheckForThreshold>
</intervalAsync>
```

The strategy forces a full index rebuild when the number of entries in the history table exceeds the number you specify in the `Indexing.FullRebuildItemCountThreshold` setting. This normally means that a substantial publishing or deployment has taken place, and this should always trigger a full index rebuild.

This behavior is only triggered when you set this property to `true` (which is the default):

```
<CheckForThreshold>true</CheckForThreshold>
```

If this setting is set to `true`, you should use the SwitchOnRebuildLuceneIndex or the SwitchOnRebuildSolrSearchIndex implementation.

The `Indexing.FullRebuildItemCountThreshold` setting is not enabled in the configuration files that Sitecore delivers. It defaults to 100000.

Attaching the IntervalAsynchronous strategy to an index

Attach this strategy to an index in the following way:

```
<index id="sitecore_index" type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex,
```

```
        Sitecore.ContentSearch.LuceneProvider">
    <param desc="name">$(id)</param>
    <param desc="folder">$(id)</param>
    <strategies hint="list:AddStrategy">
      <strategy ref="contentSearch/indexConfigurations/indexUpdateStrategies/intervalAsync" />
    </strategies>
    <Analyzer ref="search/analyzer" />
    ...
```

Best practice

Do not combine this strategy with these strategies:

- SynchronousStrategy
- OnPublishEndAsync

You can combine it with these strategies:

- RebuildAfterFullPublish
- RemoteRebuild

You should use this strategy for the master database indexes and for single-server environments where you want to use as few resources as possible.

This strategy is also useful for less critical indexes that you do not need to be updated frequently. You can adjust the interval to fit your needs.

This strategy is created for the core and master databases in the setup that Sitecore delivers:

```
<intervalAsyncCore type="Sitecore.ContentSearch.Maintenance.Strategies.
        IntervalAsynchronousStrategy, Sitecore.ContentSearch">
   <param desc="database">core</param>
   <param desc="interval">00:01:00</param>
   <CheckForThreshold>true</CheckForThreshold>
</intervalAsyncCore>
<intervalAsyncMaster type="Sitecore.ContentSearch.Maintenance.Strategies.
        IntervalAsynchronousStrategy, Sitecore.ContentSearch">
   <param desc="database">master</param>
   <param desc="interval">00:00:10</param>
   <CheckForThreshold>true</CheckForThreshold>
</intervalAsyncMaster>
```

Synchronous strategy

This strategy is the index update strategy closest to real-time. It is also the most expensive strategy in terms of CPU and I/O.

Before you use this strategy, you must be familiar with the best practices.

You specify this strategy in the following way:

```
<sync type="Sitecore.ContentSearch.Maintenance.Strategies.SynchronousStrategy, Sitecore.ContentSearch" />
```

When you attach this strategy to an index and it is initialized, you will see the following message in the `CrawlingLog` file:

```
Initializing SynchronousStrategy for index '<index_name>'.
```

When this strategy is triggered, you see this message in the `CrawlingLog` file:

```
SynchronousStrategy triggered on index '<index_name>'
```

Processing

This strategy subscribes to low-level DataEngine events, such as `ItemSaved` and `ItemSavedRemote`. When you use it on a single-server instance, it guarantees an index update immediately after an item update.

In a multiserver environment, the strategy uses the EventQueue that broadcasts remote `ItemSavedRemote` events. When an item is published and the `ItemSavedRemote` event is raised, the strategy is triggered.

Attaching the Synchronous strategy to an index

Attach this strategy to an index in the following way:

```
<index id="sitecore_index" type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex,
         Sitecore.ContentSearch.LuceneProvider">
   <param desc="name">$(id)</param>
   <param desc="folder">$(id)</param>
   <strategies hint="list:AddStrategy">
      <strategy ref="contentSearch/indexConfigurations/indexUpdateStrategies/sync" />
   </strategies>
```

```
  <Analyzer ref="search/analyzer" />
  ...
```

Best practice

Use this strategy if you need immediate index updates and you have a dedicated indexing server infrastructure that has plenty of processing resources. You should only use the Synchronous strategy on CM servers for the indexes that process the master database and where the timing of the index update is critical.

If you use this strategy on a CM server where many entries are added and changed, it can degrade system performance severely. In most cases, the IntervalAsyncronous strategy configured for the master database is sufficient.

You can only combine this strategy with the following strategy:

- RemoteRebuild

The strategy has these prerequisites:

- You must enable the EventQueue.

RemoteRebuild strategy

This strategy subscribes to the `OnIndexingEndedRemote` event. This event is triggered when a particular index is rebuilt. The strategy is only activated when a full index rebuild takes place.

You use this mechanism to rebuild remote indexes when you force an index rebuild. You specify this strategy like this:

```
<remoteRebuild type="Sitecore.ContentSearch.Maintenance.Strategies.
          RemoteRebuildStrategy, Sitecore.ContentSearch" />
```

Attaching the RemoteRebuild strategy to an index

Attach this strategy to an index in the following way:

```
<index id="sitecore_index" type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex,
         Sitecore.ContentSearch.LuceneProvider">
   <param desc="name">$(id)</param>
   <param desc="folder">$(id)</param>
   <strategies hint="list:AddStrategy">
      <strategy ref="contentSearch/indexConfigurations/indexUpdateStrategies/remoteRebuild" />
   </strategies>
   <Analyzer ref="search/analyzer" />
   ...
```

Best practice

You can combine this strategy with any other strategy. You use it in multiserver environments, where each Sitecore instance maintains its own copy of the index. You can then trigger a full rebuild from one CM server, and all remote servers where the index is configured with this strategy will rebuild.

The strategy has these prerequisites:

- The name of the index on the remote server must be identical to the name of the index that you forced to rebuild.
- You must enable the EventQueue.
- The database you assign for system event queue storage (`core` by default) must be shared between the Sitecore instance where the rebuild takes place and the other instances.

TimedIndexRefresh strategy

When you use observable crawlers, you need to consider the implications of an index operation. Observable crawlers constantly listen for new items. When a crawler receives notice of a new item, it will cache the item in memory until an index operation is processed. This has the following implications:

- Rebuilding: If you rebuild your index (clear all existing content and index new items), only the items currently held by the crawler will be inserted.
- Index Frequency: As each crawler caches each item crawled in memory, you should index at a consistent frequency to make sure that items are flushed from memory before the memory usage becomes too large.
- Update only: As your source of data is only a feed and you do not have access to all the data at once, you should only call update methods on your index.

You can use the TimedIndexRefresh strategy on an index to resolve these issues. This strategy refreshes an index with data from the crawlers, but it does not cause the index to be reset.

Configuration

Add the following to the configuration file to configure this strategy:

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
  <sitecore>
    <contentSearch>
      <configuration>
        <indexes hint="list:AddIndex">
          <index id="sitecore_atlas_index" type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex, Sitecore.ContentSearch.LuceneProvi
            <!-- ... other configuration ... -->
```

```
            <strategies hint="list:AddStrategy">
              <timed type="Sitecore.ContentSearch.Analytics.TimedIndexRefreshStrategy, Sitecore.ContentSearch.Analytics">
                <param desc="interval">00:01:00</param>
              </timed>
            </strategies>
            <locations hint="list:AddCrawler">
              <crawler type="Sitecore.ContentSearch.Analytics.AnalyticsObserverCrawler, Sitecore.ContentSearch.Analytics">
                <ObservableName>DefaultObservable</ObservableName>
                <CrawlerName>Lucene Crawler</CrawlerName>
              </crawler>
            </locations>
          </index>
        </indexes>
      </configuration>
    </contentSearch>
  </sitecore>
</configuration>
```

You must provide the interval parameter. In this example, it is configured to run every minute.

Manual strategy

This strategy disables any automatic index updates. When you use this strategy for an index, you must rebuild this index manually.

You specify this strategy like this:

```
<manual type="Sitecore.ContentSearch.Maintenance.Strategies.ManualStrategy,
          Sitecore.ContentSearch" />
```

When you attach this strategy to an index and it is initialized, you see the following message in the `CrawlingLog` file:

```
Initializing ManualStrategy for index '<index_name>'.

Index will have to be rebuilt manually
```

Attaching the Manual strategy to an index

Attach this strategy to an index in the following way:

```
<index id="sitecore_index" type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex,
         Sitecore.ContentSearch.LuceneProvider">
   <param desc="name">$(id)</param>
   <param desc="folder">$(id)</param>
   <strategies hint="list:AddStrategy">
      <strategy ref="contentSearch/indexConfigurations/indexUpdateStrategies/manual" />
   </strategies>
   <Analyzer ref="search/analyzer" />
   ...
```

Best practice

Do not combine this strategy with any other strategy. It is reserved for special situations where you have to outsource the whole indexing process to a dedicated server and you do not want any index updates on other Sitecore instances.

Send feedback about the documentation to docsite@sitecore.net.

Index-dependent HTML caching

Some index update strategies are designed to be invoked either when publishing completes or after a specific interval of time has passed. If your Sitecore instance uses HTML caching for renderings, controls, or sublayouts that contain code, and this code depends on an index, a race condition between HTML cache clearing and index update operations can occur.

You can solve this issue by selecting `Cacheable` and `Clear on Index Update` for these renderings, controls, and sublayouts:

You can do this on the rendering definition item or on the layout details for a specific content item, in a similar way to how you can set the `Vary by` settings

When you select `Clear on Index Update`, Sitecore clears the HTML cache of renderings when the index is updated, using the IndexDependentCacheManager component. The execution of this component is triggered from the `indexing:end` and `indexing:end:remote` events in the `Sitecore.ContentSearch.config` file:

```
<event name="indexing:end">
    <handler type="Sitecore.ContentSearch.Maintenance.
        IndexDependentHtmlCacheManager, Sitecore.ContentSearch" method="Clear" />
</event>
<event name="indexing:end:remote">
    <handler type="Sitecore.ContentSearch.Maintenance.
        IndexDependentHtmlCacheManager, Sitecore.ContentSearch" method="Clear" />
</event>
```

Send feedback about the documentation to docsite@sitecore.net.

Multiple indexes (sharding)

Index sharding is a process that splits the documents in an index into smaller partitions. These smaller partitions are called *shards*. The result is that instead of all documents being in one large index, documents are distributed between shards. The sharding process has logic (the "sharding strategy") that decides how the documents are allocated to the shards.

The basic sharding process is this:

One index satisfies the needs of most Sitecore solutions but multiple indexes offer better scaling when needed.

Sharding and Solr

When you use Solr, Sitecore does not handle the sharding. Instead, the SolrCloud feature of the Solr application handles the sharding.

Solr can automatically assign documents to shards (similar to what Sitecore can do for Lucene) and it has extra features, such as replicated shards. Replicated shards are useful for handling failure and failover scenarios.

The Sitecore implementation of Solr handles a sharded endpoint in the same way it handles an unsharded endpoint. You do not need any extra configuration to work with Solr sharded indexes.

Note

Sitecore does not fully support failover. Specifically, Sitecore (as a Solr client) cannot switch between Solr servers (Solr replicas) if the current server (leader) goes down.

For more information about the configuration of the SolrCloud, go to https://cwiki.apache.org/confluence/display/solr/SolrCloud

Sharding and Lucene

When you use Lucene, the data from each of the three Sitecore databases (master, web, and core) is, by default, stored in a single search index. As your search index grows, you can implement a sharding strategy to store the data from each database in its own separate search index.

You can also shard in other ways. For example, you can have a separate index for the media library.

If you use buckets and have thousands or millions of items, sharding is an approach you can use if you want to continue using Lucene. If your search indexes continue to grow and become too large for this strategy, you should switch to using Solr.

Note

If you use sharding, you must turn off the other Lucene configuration files because leaving these enabled will create redundant indexes.

Configure multiple search indexes

Sitecore provides the following example configuration files that help you create an index for each database:

`Sitecore.ContentSearch.Lucene.Indexes.Sharded.Core.config.example`

`Sitecore.ContentSearch.Lucene.Indexes.Sharded.Master.config.example`

`Sitecore.ContentSearch.Lucene.Indexes.Sharded.Web.config.example`

These files are stored in the *Include* folder (`wwwroot\<site name>\Website\App_Config\Include`).

If these configuration files are not sharded enough, you can change the configuration to fit your needs.

Use the following code sample and table to see what you need to add:

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
  <sitecore>
    <contentSearch>
      <configuration type="Sitecore.ContentSearch.LuceneProvider.LuceneSearchConfiguration,
       Sitecore.ContentSearch.LuceneProvider">
        <indexes hint="list:AddIndex">
          <index id="sitecore_core_index"
           type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex,
            Sitecore.ContentSearch.LuceneProvider">
            <param desc="name">$(id)</param>
            <param desc="folder">$(id)</param>
            <!-- This initializes index property store. Id has to be set to the index id -->
            <param desc="propertyStore" ref="contentSearch/databasePropertyStore"
             param1="$(id)" />
            <strategies hint="list:AddStrategy">
              <!-- NOTE: order of these is controls the execution order -->
              <strategy ref="contentSearch/indexUpdateStrategies/intervalAsyncCore" />
            </strategies>
            <commitPolicy hint="raw:SetCommitPolicy">
              <policy type="Sitecore.ContentSearch.TimeIntervalCommitPolicy,
               Sitecore.ContentSearch" />
            </commitPolicy>
            <commitPolicyExecutor hint="raw:SetCommitPolicyExecutor">
              <policyExecutor type="Sitecore.ContentSearch.CommitPolicyExecutor,
               Sitecore.ContentSearch" />
            </commitPolicyExecutor>
            <locations hint="list:AddCrawler">
              <crawler type="Sitecore.ContentSearch.LuceneProvider.Crawlers.DefaultCrawler,
               Sitecore.ContentSearch.LuceneProvider">
                <Database>core</Database>
                <Root>/sitecore</Root>
              </crawler>
            </locations>
          </index>
        </indexes>
      </configuration>
    </contentSearch>
  </sitecore>
</configuration>
```

Name	Description	Example
`<Root>`	Specify the root node of the content tree to be included in the index.	`<Root>/sitecore/media library</Root>`
`<name>`	Name of the search index.	`<param desc="name">$(id)</param>`
`<Database>`	Database name.	`<Database>core</Database>`

`<strategies>`	List of index strategies to run.	`<strategies hint="list:AddStrategy">`
		`<strategy`
		`ref="contentSearch/indexUpdateStrategies/intervalAsyncCore" />`
		`</strategies>`

<CommitPolicy>	Controls when the index commits what it has in memory or in temporary files to disk. This can be time based or document count based.	<commitPolicy hint="raw:SetCommitPolicy"> <policy type="Sitecore.ContentSearch.TimeIntervalCommitPolicy,Sitecore.ContentSearch"/> </commitPolicy>
<commitPolicyExecutor>	The class that executes the commit.	<commitPolicyExecutor hint="raw:SetCommitPolicyExecutor"> <policyExecutor type="Sitecore.ContentSearch.CommitPolicyExecutor, Sitecore.ContentSearch" /> </commitPolicyExecutor>

Index context switcher

If you use sharding, Sitecore uses the `<Root>` element in relation to the `Context.Item` to determine which index to use. This index switching is automatic.

Note

The more specific your `<Root>` is, the higher it needs to be listed in the configuration file. The index context switcher uses the indexes in the order that they are listed.

For example, if you have an index `<Root>` element of `/sitecore/content/Home`, it should be located below the index for a `<Root>` element of `/sitecore/content/Home/Flights`:

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
  <sitecore>
    <contentSearch>
      <configuration type="Sitecore.ContentSearch.LuceneProvider.LuceneSearchConfiguration,
       Sitecore.ContentSearch.LuceneProvider">
        <indexes hint="list:AddIndex">
          <index id="sitecore_core_index"
           type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex,
            Sitecore.ContentSearch.LuceneProvider">
            <param desc="name">$(id)</param>
            <param desc="folder">$(id)</param>
            <!-- This initializes index property store. Id has to be set to the index id -->
            <param desc="propertyStore" ref="contentSearch/databasePropertyStore"
             param1="$(id)" />
            <strategies hint="list:AddStrategy">
              <!-- NOTE: order of these is controls the execution order -->
              <strategy ref="contentSearch/indexUpdateStrategies/intervalAsyncCore" />
            </strategies>
            <commitPolicy hint="raw:SetCommitPolicy">
              <policy type="Sitecore.ContentSearch.TimeIntervalCommitPolicy,
               Sitecore.ContentSearch" />
            </commitPolicy>
            <commitPolicyExecutor hint="raw:SetCommitPolicyExecutor">
              <policyExecutor type="Sitecore.ContentSearch.CommitPolicyExecutor,
               Sitecore.ContentSearch" />
            </commitPolicyExecutor>
            <locations hint="list:AddCrawler">
              <crawler type="Sitecore.ContentSearch.LuceneProvider.Crawlers.DefaultCrawler,
               Sitecore.ContentSearch.LuceneProvider">
                <Database>core</Database>
                <Root>/sitecore</Root>
              </crawler>
            </locations>
          </index>
        </indexes>
      </configuration>
    </contentSearch>
  </sitecore>
</configuration>
```

Default sharding strategy

Sitecore provides a default sharding strategy called the `LucenePartitionShardingStrategy`. This strategy takes a document and calculates a hash of the ID to determine which shard to put it into. This hashing is very fast and does not rely on any shared state or ID generation. This approach does not give you a completely even distribution (for example, 100 documents are not split 50/50) but it improves performance considerably.

This strategy only has one option: the `shardDistribution` parameter. You must set this parameter to be a factor of 2 (2, 4, 8, 16, …) and this specifies how many shards the index is split into.

Create a new sharding strategy

If the default strategy is not what you need, you can implement your own strategy. You do this by using the `Sitecore.ContentSearch.Sharding.IShardingStrategy` interface, and passing the implementation into the index.

You should rebuild your index after applying a strategy. It is not essential, but it will give the index a more even distribution of documents.

Send feedback about the documentation to docsite@sitecore.net.

Rebuild search indexes

There are situations where you need to rebuild indexes, for example, when you deploy to a production environment, when indexes are out of date, or when indexes have been corrupted.

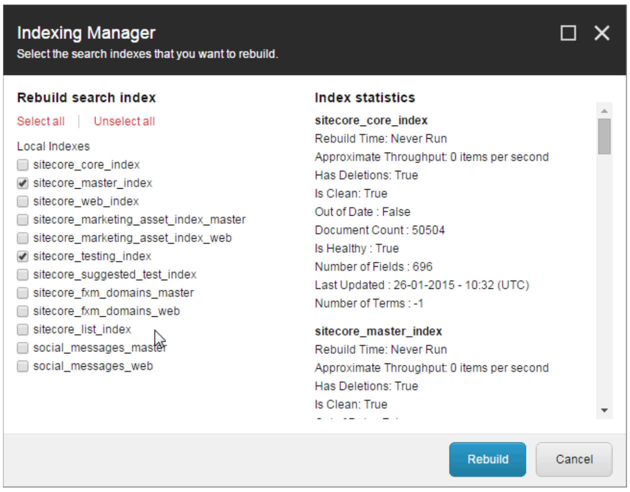
There are three different ways to rebuild indexes. This topic describes how to:

- Use the Sitecore Control Panel
- Use custom code
- Use the Content Editor

Use the Sitecore Control Panel

To rebuild indexes from the Sitecore Control Panel:

1. Log in to the Launchpad.
2. Open the Control Panel.
3. In the Indexing section, click Indexing Manager.
4. In the Indexing Manager dialog box, select the indexes you want to rebuild, and click Rebuild:



Use custom code

You can rebuild indexes from custom code by running a script similar to one of these from a custom `.aspx` page:

```
// To rebuild "new" search indexes, use this piece of code for every "new" index

IndexCustodian.FullRebuild(ContentSearchManager.GetIndex("[INDEX NAME]"), true);

// Or to rebuild all indexes, use the following piece of code:

IndexCustodian.RebuildAll();
```

Use the Content Editor

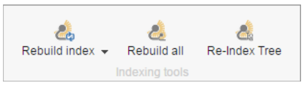
To rebuild indexes from the Content Editor:

1. In the Content Editor, click the Developer tab.

   Note

   If you do not see the Developer tab, right-click in the menu bar and select *Developer* in the list of tabs.

2. In the Indexing tools group, select one of the options to rebuild indexes:



- Rebuild index – opens a list of all indexes. This list contains information about when each index was last rebuilt. Select an index to rebuild it.
- Rebuild all – rebuilds all indexes.
- Re-Index Tree – process the selected item and all descendants recursively, and force an update operation on all indexes that are related to the selected content area.

  Note

  Use this command only in situations where incremental indexing is either not working as expected, or when you use the manual update strategy. You should only do this when absolutely necessary.

Send feedback about the documentation to docsite@sitecore.net.

Switch Lucene indexes

You can set up Lucene to maintain two directories for an index. This addresses the fact that the index directory is reset (deleted) before a full index rebuild. This is only a problem for production environments, so you can reconfigure your custom index using the *SwitchOnRebuildLuceneIndex* class during testing and before moving to a production environment.

The *SwitchOnRebuildLuceneIndex* class inherits from the *LuceneIndex* class. It adds the capability of maintaining two directories for a particular index.

When you initialize a *SwitchOnRebuildLuceneIndex* object, it verifies the presence of a secondary folder. If the folder does not exist, it creates this folder. It appends `_sec` to the name of the baseline directory, and uses this as the name of folder it creates.

The index uses the `IndexReader.LastModified(Directory)` method to choose a primary directory based on the last modified date. It uses the most recent directory as the primary directory.

The [Index Property Store]() contains information about which directory is primary and which one is secondary.

The `SwitchOnRebuildLuceneIndex` object writes the following entries to the `CrawlingLog` object when it initializes:

```
“Resolving directories from index meta data store”
"Resolving directories by last time modified"
"Primary directory last modified = '…'"
"Secondary directory last modified = '…'"
“ReadUpdateDirectory is set to …”
“FullRebuildDirectory is set to '{0}'”
```

When the full index rebuild completes, the primary and secondary directories are switched.

To set up Lucene to maintain two directories for an index:

- Change the type reference on a particular search index to `Sitecore.ContentSearch.LuceneProvider.SwitchOnRebuildLuceneIndex`:

```
<indexes hint="list:AddIndex">
   <index id="content_index"
          type="Sitecore.ContentSearch.LuceneProvider.SwitchOnRebuildLuceneIndex,
                Sitecore.ContentSearch.LuceneProvider">

        <param desc="name">$(id)</param>
        <param desc="folder">$(id)</param>
...
```

Note

After you have adjusted the configuration file to make the search index use the `SwitchOnRebuildLuceneIndex` method, your website uses indexes from the primary directory. Each time you do a full index rebuild, this rebuild take place in the secondary directory. The secondary directory becomes the primary one after the rebuild.

Send feedback about the documentation to docsite@sitecore.net.

Switch Solr indexes

You can set up Solr to rebuild an index in a separate core so that the rebuilding does not affect the search index that is currently used. Once the rebuilding and the optimization of the index completes, Sitecore switches the two cores, and the rebuilt and optimized index is used.

The *SwitchOnRebuildSolrSearchIndex* class inherits from the *SolrSearchIndex* class and adds the capability of maintaining two cores for a particular index. Because this is only important for production environments, you can reconfigure your custom index with the *SwitchOnRebuildSolrSearchIndex* implementation during testing and before moving to a production environment.

Note

The primary core is used for index read and update operations. The secondary core is a fallback for read operations during a full index rebuild.

When it has fully rebuild the index, Sitecore switches the primary and secondary cores by sending a [SWAP]() request to Solr.

To create a duplicate Solr index and set up Solr to rebuild an index in a separate core:

1. From the Solr server, copy the existing *itembuckets* folder. Call the copy `itembuckets_sec`.
2. Update the `solr.xml` file like this:

```
<cores defaultCoreName="itembuckets" adminPath="/admin/cores" zkClientTimeout="${zkClientTimeout:15000}" hostPort="8983" hostConte
<core loadOnStartup="true" instanceDir="itembuckets\" transient="false" name="itembuckets"/>
<core loadOnStartup="true" instanceDir="itembuckets_sec\" transient="false" name="itembuckets_sec "/>
 </cores>
```

3. Restart Solr.
4. Check the two cores. You can do this by visiting these URLs (change as appropriate for your actual setup):

```
http://localhost:8983/solr/itembuckets/select/?q=*:*&version=2.2&start=0&rows=10&indent=on
http://localhost:8983/solr/itembuckets_sec/select/?q=*:*&version=2.2&start=0&rows=10&indent=on
```

   Both should return 0 results (but you will see some XML).

5. To use this implementation, change the type reference on a particular search index to `Sitecore.ContentSearch.SolrProvider.SwitchOnRebuildSolrSearchIndex`, and add the `rebuildcore` parameter:

```
<indexes hint="list:AddIndex">
   <index id="content_index"
          type="Sitecore.ContentSearch.SolrProvider.SwitchOnRebuildSolrSearchIndex,
                Sitecore.ContentSearch.SolrProvider">

       <param desc="name">$(id)</param>
        <param desc="core">itembuckets</param>
  <param desc="rebuildcore">itembuckets_sec</param>
...
```

Note

After you have changed the configuration file, your website uses indexes from the primary core. Each time you initiate a full index rebuild, Sitecore does this in the secondary core. The secondary core then becomes the primary one after the rebuild.

Separate cores for each index

If you configure multiple indexes to share the same core, then the secondary core becomes the primary for *all* indexes each time you perform a full index rebuild.

For example, assume that you have a configuration similar to this:

```
<index id="sitecore_web_index" type="Sitecore.ContentSearch.SolrProvider.SwitchOnRebuildSolrSearchIndex, Sitecore.ContentSearch.SolrPro
    <param desc="name">$(id)</param>
    <param desc="core">itembucket</param>
    <param desc="rebuildcore">itembucket_web_rebuild</param>
...
<index id="sitecore_master_index" type="Sitecore.ContentSearch.SolrProvider.SolrSearchIndex, Sitecore.ContentSearch.SolrProvider">
    <param desc="name">$(id)</param>
    <param desc="core">itembucket</param>
    <param desc="propertyStore" ref="contentSearch/databasePropertyStore" param1="$(id)" />
...
```

When Sitecore has finished rebuilding the index, both indexes use the `itembucket_web_rebuild` core, even if you have not specified the *SwitchOnRebuild* behavior for the `sitecore_master_index` index.

For this reason, you should configure separate cores for each index in production. You can do this by using configuration similar to this:

```
<index id="sitecore_web_index" type="Sitecore.ContentSearch.SolrProvider.SwitchOnRebuildSolrSearchIndex, Sitecore.ContentSearch.SolrPro
    <param desc="name">$(id)</param>
    <param desc="core">$(id)</param>
    <param desc="rebuildcore">$(id)_rebuild</param>
...
<index id="sitecore_master_index" type="Sitecore.ContentSearch.SolrProvider.SolrSearchIndex, Sitecore.ContentSearch.SolrProvider">
    <param desc="name">$(id)</param>
    <param desc="core">$(id)</param>
    <param desc="propertyStore" ref="contentSearch/databasePropertyStore" param1="$(id)" />
...
```

This helps you avoid unclear behaviors involving *SwitchOnRebuild* and cache clearing.

Send feedback about the documentation to docsite@sitecore.net.

The index property store

The index property store persists some metadata for search indexes. For example, the last updated timestamp is in the index property store.

You configure the index property store in the `sitecore/contentSearch` part of the `Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config` file. The configuration delivered with Sitecore looks like this:

```
<databasePropertyStore
          type="Sitecore.ContentSearch.Maintenance.IndexDatabasePropertyStore,
          Sitecore.ContentSearch">
            <Key>$(1)</Key>
            <Database>core</Database>
</databasePropertyStore>
```

This index property store configuration uses the *Properties* table in the Sitecore database as the persistent store, and it uses the `core` database.

You assign the index property store to each index with this configuration:

```
<index id="content" type="Sitecore.ContentSearch.LuceneProvider.LuceneIndex,
           Sitecore.ContentSearch.LuceneProvider">
                  <param desc="name">$(id)</param>
                  <param desc="folder">$(id)</param>
                  <param desc="propertyStore"
                     ref="contentSearch/databasePropertyStore" param1="$(id)" />
```

You must set the first `param1` attribute to `$(id)` or something similar. It must be unique across all indexes.

The index property store is a key/value-based storage. The key consists of a master key and the key of the record that has to be saved.

The master key depends on the configuration. In the example above, the `Key` parameter of the `databasePropertyStore` definition node (`<Key>$(1)</Key>`) will pick up the first parameter from the place where `databasePropertyStore` is referenced:

```
<param desc="propertyStore" ref="contentSearch/databasePropertyStore" param1="$(id)" />
```

Because this record is defined in the `<index />` element, the value of `$(id)` will be `content`:

```
<index id="content">
```

This is how the `databasePropertyStore` guarantees that keys are unique across all indexes. It also appends the `InstanceName` to the master key, and this guarantees that keys are unique across all environments.

With the configuration described above, you can save a custom property to the index property store with this code:

```
index.PropertyStore.Set("docs-count", "123");
```

The value `123` is saved with a key similar to this: `content_CM01_docs-count`, where `content` is the index ID, and `CM01` is the name of the Sitecore instance.

A custom index property store

If you cannot use the implementation delivered by Sitecore, you can create a custom index property store implementation by implementing the `IIndexPropertyStore` interface.

When you have implemented this interface, you need to define it in the configuration and assign it to indexes. You must define it in the `<contentSearch />` part of the configuration, for example:

```
<contentSearch>
         <fileSystemPropertyStore type="Custom.IndexFileSystemPropertyStore, Custom">
            <Key>$(1)</Key>
            <FilePath>$(dataFolder)/indexpropertystore</FilePath>
         </fileSystemPropertyStore>
...
<index id="content" type="Sitecore.ContentSearch.LuceneProvider.
          LuceneIndex, Sitecore.ContentSearch.LuceneProvider">
                  <param desc="name">$(id)</param>
                  <param desc="folder">$(id)</param>
                  <param desc="propertyStore"
                           ref="contentSearch/fileSystemPropertyStore"
                           param1="$(id)"/>
```

Send feedback about the documentation to docsite@sitecore.net.

The search and indexing log files

Sitecore uses log files for tracking search and indexing operations. The crawling log shows information about indexing operations, and the search log shows information about search queries.

Crawling log

The crawling log contains information about the indexing process. You set up the crawling log in the `<log4net />` section of the `web.config` file. The default logging level is `INFO`:

```
<logger name="Sitecore.Diagnostics.Crawling" additivity="false">
      <level value="INFO" />
      <appender-ref ref="CrawlingLogFileAppender" />
</logger>
```

You define the `appender` for this log in the same section. By default, it writes to a .txt file in the *data/logs* folder:

```
<appender name="CrawlingLogFileAppender" type=
          "log4net.Appender.SitecoreLogFileAppender, Sitecore.Logging">
      <file value="$(dataFolder)/logs/Crawling.log.{date}.txt" />
      <appendToFile value="true" />
      <layout type="log4net.Layout.PatternLayout">
        <conversionPattern value="%4t %d{ABSOLUTE} %-5p %m%n" />
      </layout>
</appender>
```

The implementation uses [log4net](). The log4net documentation describes how you can change the `appender` to write to a Windows Event Log, a database, or any other location.

This is an example of crawling log entries:

```
INFO  [Index=sitecore_core_index] Initializing IntervalAsynchronousUpdateStrategy
              with interval '00:01:00'.
INFO  [Index=sitecore_core_index] Initializing LuceneDatabaseCrawler.
              DB:core / Root:/sitecore
INFO  [Index=sitecore_master_index] Initializing SynchronousStrategy.
INFO  [Index=sitecore_master_index] Initializing LuceneDatabaseCrawler.
              DB:master / Root:/sitecore
INFO  [Index=sitecore_web_index] Initializing OnPublishEndAsynchronousStrategy.
INFO  [Index=sitecore_web_index] Initializing LuceneDatabaseCrawler.
              DB:web / Root:/sitecore
INFO  [Index=custom_master] Initializing IntervalAsynchronousUpdateStrategy
              with interval '00:00:05'.
INFO  [Index=custom_master] Initializing LuceneDatabaseCrawler. DB:master /
              Root:{D70CBEED-6DCF-483F-978F-6FC3C8049512}
INFO  [Index=custom_web] Initializing OnPublishEndAsynchronousStrategy.
INFO  [Index=custom_web] Initializing LuceneDatabaseCrawler.
              DB:web / Root:{D70CBEED-6DCF-483F-978F-6FC3C8049512}
INFO  [Index=custom_web] Creating primary and secondary directories
INFO  [Index=custom_web] Resolving directories from index property
              store for index 'custom_web'
INFO [Index=custom_master] IntervalAsynchronousUpdateStrategy executing.
```

The log begins with information about how indexes are configured and initialized. After that, an entry is written in the crawling log when an index update strategy is applied to an index or when a full rebuild is triggered.

Because the logging level in the example is `INFO`, there is a limited amount of entries in the log file.

For more detailed logs, for example, for troubleshooting purposes, you can change the logging level to `DEBUG`:

```
<logger name="Sitecore.Diagnostics.Crawling" additivity="false">
      <level value="DEBUG" />
      <appender-ref ref="CrawlingLogFileAppender" />
</logger>
```

With this configuration, and with the verbose logger enabled, Sitecore produces a very detailed item level indexing log.

Verbose logging

You can enable verbose logging in addition to the standard logging. When you enable verbose logging, you can see more detail about the indexing.

You use verbose logging in situations such as:

- An index is not updated.
- Some items are not being indexed.
- A full rebuild is triggered for an index and you need to understand why.
- You want to explore indexing activity on a specific server.

You want to test how your index is updated across all servers in a multiserver environment

The *VerboseLogger* is instantiated when `ContentSearch.VerboseLogging` is `true`:

The default is `false` and the setting is not present in the configuration delivered by Sitecore. The `Sitecore.ContentSearch.VerboseLogging.config.example` file has an example that shows you how to enable verbose logging.

Note

You must only enable the `VerboseLogger` component in special circumstances and never run it for long periods in a production environment. The log file grows very fast, and this can degrade performance.

You use the verbose logging for troubleshooting. For example, if a specific item is not indexed, the `VerboseLogger` gives you more context and can help you solve the problem.

The `VerboseLogger` writes information about a large set of indexing events:

- `indexing:excludedfromindex`
- `indexing:start`
- `indexing:end`
- `indexing:addingrecursive`
- `indexing:addedrecursive`
- `indexing:adding`
- `indexing:added`
- `indexing:refreshstart`
- `indexing:refreshend`
- `indexing:deleteitem`
- `indexing:deletegroup`
- `indexing:updatingitem`
- `indexing:updateditem`
- `indexing:updatedependents`
- `indexing:refreshstart`
- `indexing:refreshend`
- `indexing:propertyset`
- `indexing:propertyget`
- `indexing:propertyadd`

Search log

The search log contains information about the search queries that Sitecore executes.

You set up the search log in the `<log4net />` section of the `web.config` file:

```
<logger name="Sitecore.Diagnostics.Search" additivity="false">
  <level value="INFO" />
  <appender-ref ref="SearchLogFileAppender" />
</logger>
```

You define the `appender` for this log in the same section. By default, it is set up to write to a .txt file in the *data/logs* folder:

```
<appender name="SearchLogFileAppender"
           type="log4net.Appender.SitecoreLogFileAppender, Sitecore.Logging">
      <file value="$(dataFolder)/logs/Search.log.{date}.txt" />
      <appendToFile value="true" />
      <layout type="log4net.Layout.PatternLayout">
        <conversionPattern value="%4t %d{ABSOLUTE} %-5p %m%n" />
      </layout>
</appender>
```

The implementation is based on [log4net](). The log4net documentation describes how you change the `appender` to write to a Windows Event Log, a database, or any other location.

This is a sample of what the search log entries look like:

```
3212 19:31:56 INFO  ExecuteQueryAgainstLucene :
         +_datasource:sitecore +(+(+_path:11111111111111111111111111111111 +_latestversion:1)
         +mileagehwy:[1 TO 4mileagecity]) - Filter : 3212 19:31:56
       INFO  Results from web database :8818
```

You can enable a full level debug of content searches by setting `ContentSearch.EnableSearchDebug = true in the Sitecore.ContentSearch.config` file and changing the logging level for the search logger to `DEBUG`:

```
<logger name="Sitecore.Diagnostics.Search" additivity="false">
      <level value="DEBUG" />
      <appender-ref ref="SearchLogFileAppender" />
</logger>
```

The entries in the search are useful for:

- Developers who want to understand how the Linq code translates into the native search queries that are passed on to the search provider (Lucene or Solr).
- Administrators who want to understand which search queries are performed on a specific server. They can use this information for further optimization, for example, as input to which queries they should put into the queryWarmup pipeline.

Send feedback about the documentation to docsite@sitecore.net.

Using Solr or Lucene

Sitecore supports both Lucene and Solr search engines. The search engines are used for searching in the content databases, as well as for searching in a number of operational databases that Sitecore uses for collecting analytics data, test data, and so forth.

When visitors search for content, they use a search engine. Therefore, this search engine must be available from the Internet. On the other hand, the search engine that Sitecore uses for the operational databases does not have to be available from the Internet.

You must use Solr if you have a scaled environment.

Solr supports calls over HTTP(S) which means that the indexes are available to all servers in the environment that require it (content management and processing servers).

Lucene is a file based indexing system, which means if an index is not located on the server that the request is coming from you have to ensure that indexes across all servers remain in sync. This is technically possible but non-trivial, and not supported.

Other reasons for using Solr are:

- When you need to index large numbers of items (50,000 and up), Solr performs better.
- Solr is more robust. If your site depends on search as the primary interface, we recommend using Solr.

You can move from Lucene to Solr. The LINQ queries remain the same, but there are differences in configuration and so forth that you need to address. You will also have to rebuild indexes. The move is possible, but it is not a trivial task. You should therefore use Solr if you plan to scale your site (have a distributed setup with multiple servers).

If you do move from Lucene to Solr, be aware that Lucene and Solr score search results in slightly different ways (which can look like there is something wrong unless you know this).

You can mix Lucene and Solr, and, for example, use Solr for xDB and Lucene for content search at the same time. Generally it is simpler to use Solr for everything once you need to use Solr for the operational databases.

Send feedback about the documentation to docsite@sitecore.net.

Security and item buckets

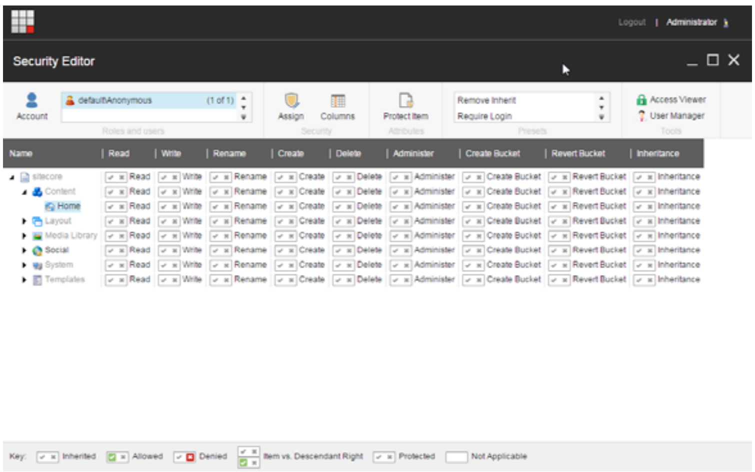
An item bucket removes the hierarchy of items for the items that it contains, but inheritance is still in effect for security settings.

Sitecore implements inheritance when content is retrieved. If a user does not have access at either the user/role level or at the item/ancestor level, an item is not retrieved.

Security is not used for the number of items shown for a search facet, but the item security still applies. This means that users will sometimes see a number of items in a facet, but they will not have access to all these items.

You can restrict the ability of a user or role to convert a content item into an item bucket or to convert an item bucket back into a normal item in the Security Editor. The Sitecore security system has two access rights settings for item buckets:

- Create Bucket
- Revert Bucket



Note

If these security settings are not immediately available in the Security Editor, in the Security group, click Columns and then in the Columns dialog box, select Create Bucket and Revert Bucket.

Send feedback about the documentation to docsite@sitecore.net.

Sitecore security

The Sitecore security model enables you to grant or deny access to almost every aspect of a website. To do this, you use security accounts and security domains to control the access that users have to the items and content on their website as well as the access they have to Sitecore functionality.

Security accounts

A security account is a role or a user:

- A role is a collection of users or a collection of users and other roles. You can use roles to assign access rights to groups of Sitecore users by making them a member of a role.

  Sitecore contains a set of predefined security roles that you can use, or you can create new roles and give them the relevant access rights.

  Note

  In Sitecore, there are also global roles that all users across domains can see. These roles are listed in the `/App_Config/Security/GlobalRoles.config` file.

- A user account in Sitecore contains details about the user name, domain, email, and password. You can assign access rights directly to a user account.

Note

In Sitecore, a security account is identified by its name (domain name\account name, for example, sitecore\Developer or sitecore\ECM advanced users) and therefore, two security accounts cannot have the same name.

Security domains

A security domain is a collection of security accounts (users and roles) that you can administer as a unit with common rules and procedures. A domain is used to collect security accounts that have some logical relationship, for example, all the accounts that have access to use the Sitecore clients could be stored in the Sitecore domain, whereas all the accounts with access to the published website could be stored in the Extranet domain.

The security engine

Sitecore uses the .NET security engine, which offers several advantages:

- A variety of plug-and-play features provided directly by Microsoft.
- An abstraction from the real data source.
- The option to replace or extend the default configuration with custom providers.
- The performance speed of a pure ASP.NET solution.
- The possibility of keeping the accounts in identifiable storage areas by using several providers simultaneously.

Send feedback about the documentation to docsite@sitecore.net.

The security tools

In Sitecore, there are several tools available that you can use to manage various aspects of security.

The Sitecore security tools are:

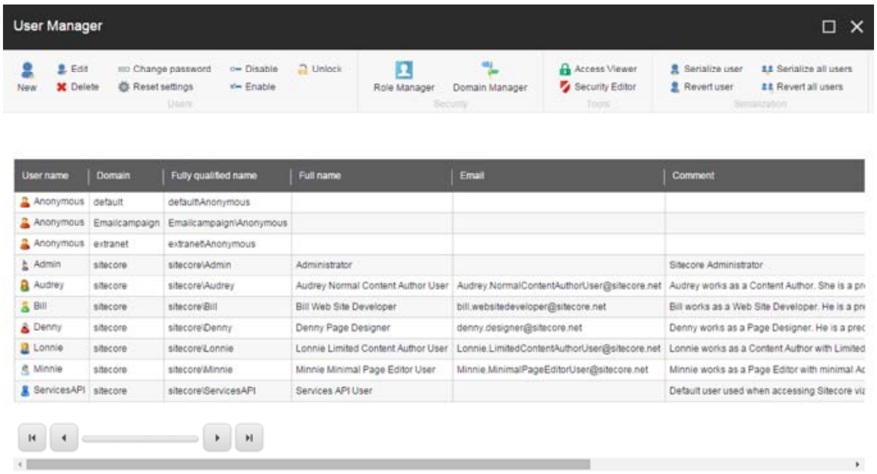
- The User Manager
- The Role Manager
- The Security Editor
- The Access Viewer
- The Domain Manager
- The Content Editor — the Security tab

The User Manager

You can use the User Manager to create and manage the users that have access to your system.

In the User Manager, you can:

- Create, edit, and delete users
- Change the password of users
- Enable and disable users
- Lock and unlock users
- Open the other security tools

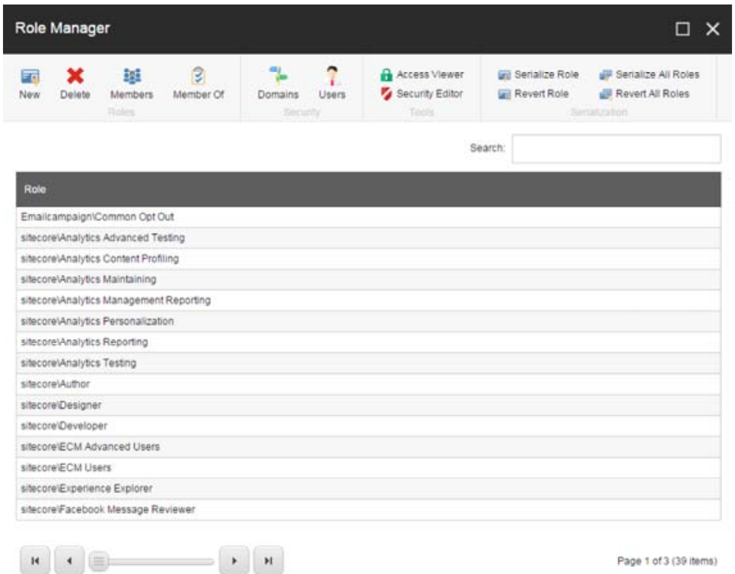


The Role Manager

You can use the Role Manager to create and manage the roles that you want to assign to your security accounts (users and roles).

In the Role Manager, you can:

- Create and delete roles.
- Add or remove users and roles as members of a role.
- Open the other security tools.

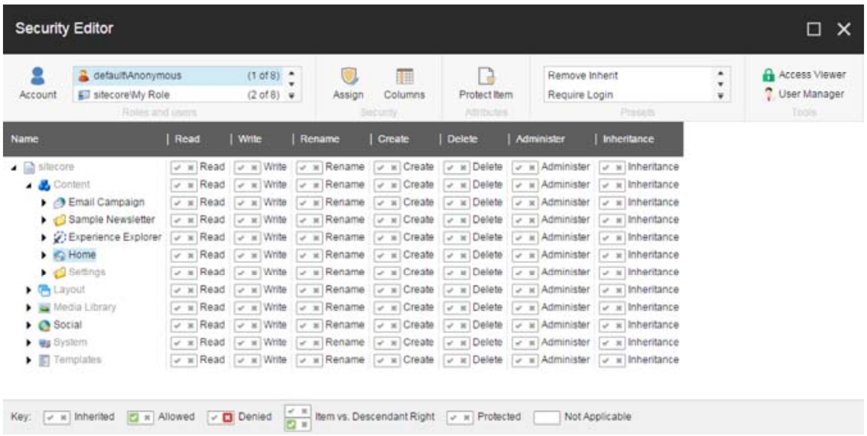


The Security Editor

You can use the Security Editor to assign the access rights that roles and users should have to the items in the content tree.

In the Security Editor, you can:

- Assign access rights to your security accounts
- Protect and unprotect items
- Open the Access Viewer and the User Manager

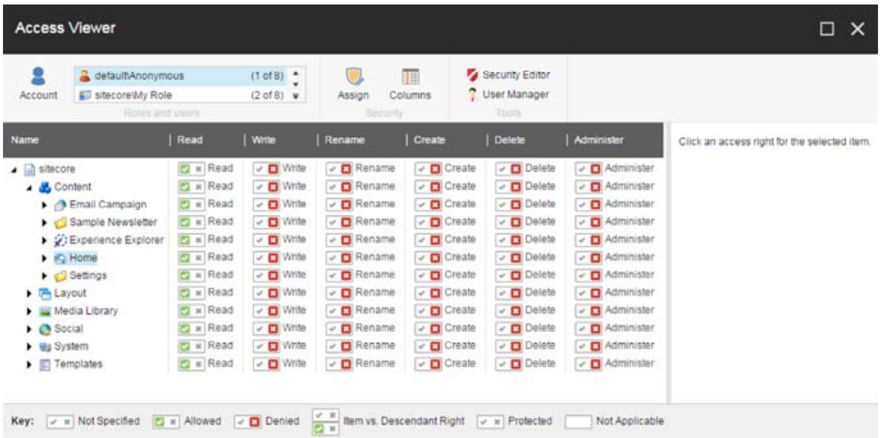


The Access Viewer

You can use the Access Viewer to get an overview of the access rights that are assigned to your security accounts.

In the Access Viewer, you can:

- Get an overview of the access rights that are assigned to the security accounts for each item in the content tree.
- See an explanation of how the current settings have been resolved.
- Open the Security Editor and the User Manager.

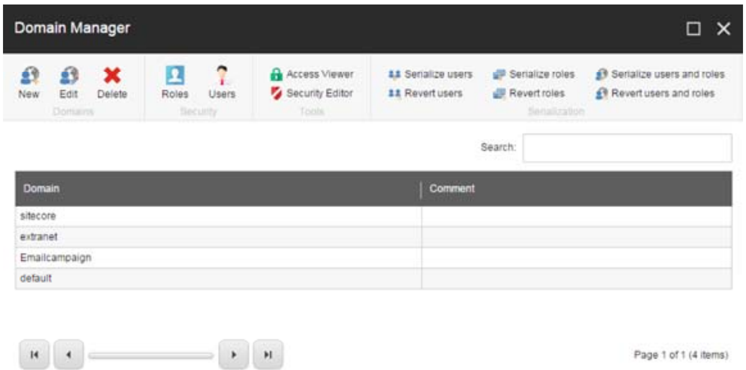


The Domain Manager

You can use the Domain Manager to create and manage your domains.

In the Domain Manager, you can:

- Create and delete domains.
- Specify whether the domains are global or locally managed.
- Open other security tools.



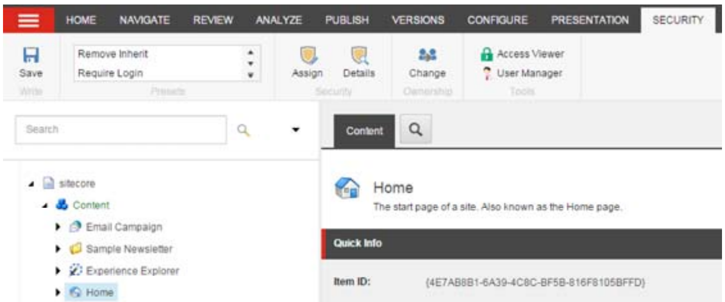
The Content Editor — the Security tab

In the Content Editor, there are security functionalities available that enable you to assign access rights to your security accounts.

In the Content Editor, on the Security tab, you can:

- Assign access rights to your security accounts.

- Get an overview of the roles and users that have access rights to individual items in the content tree.
- Change the ownership of an item.
- Open the Access Viewer and the User Manager.



Send feedback about the documentation to docsite@sitecore.net.

Assign access rights to a security account

You assign access rights to a security account on a per item basis. However, an item always inherits the access rights that is assigned to its parent item. This makes it easier for you to manage your security system because you do not necessarily have to assign access rights for each item in your content tree. Instead, you only need to assign and manage the access rights on the parent items and then specify whether their descendants can inherit these access rights.

This topic outlines how to:

- Assign access rights to a security account
- Assign access rights explicitly on an item
- Assign access rights to multiple items

Assign access rights to a security account

To assign access rights to a role or a user:

1. On the Sitecore Launchpad, click Security Editor.

   cms_LaunchPad_SecurityEditor

2. In the Security Editor, in the Roles and users group, click the user or role that you want to assign the access rights to, for example *My Role*.

   cms_all_SecurityEditor

   Note

   If the relevant account is not available in the field, click Account, and then in the Add an Account dialog box, navigate to the relevant user or role, and click OK.

3. In the content tree, click the item that you want the security account to have access rights to.

4. For each type of access right to the item (*Read, Write, Rename, Create, Delete*, *Administer, and Inheritance)* click to grant or click

   to deny the security account the access right to the item.

   Note

   The Sitecore/Everyone role is assigned to all users and roles, which means that they have *Read* access to all items by default.

5. In the ribbon, in the Attributes and Presets groups, specify additional security rights to the item:
   - Remove Inherit – this sets the *Inherit* right to *Denied* for the *'*' (Everyone)* role. In this way, the item or its descendants do not inherit any security rights from any parent items.
   - Require Login – this sets the *Read* access to *Denied* for the *extranet/Anonymous* user. In this way, you can require extranet login to access the item and its descendants.
   - Protect Item – this makes the item unavailable for editing. For example, you can protect an item if you want to make sure that a user cannot accidentally delete an important item. Only an administrator or a user that is a member of the *Sitecore Client Configuring* role can protect an item and unprotect the item again.
6. Verify your setting in the Access Viewer. In the Security Editor, in the Tools group, click Access Viewer.

   Note

   Remember the user can be a member of multiple roles, which may have access rights to the item that overrule the access rights that you have just assigned.

Assign access rights specifically to an item

In the Security Editor, you can specifically deny a security account the access to an item in the content tree. In this way, you can overrule the access rights that the item otherwise inherits from its parent item.

To deny a role access rights specifically to an item:

1. In the Security Editor, in the Roles and users group, click the role or user that you want to grant or deny access to.

   Note

   If the relevant account is not available in the field, click Account, and then in the Add an Account dialog box, navigate to the relevant user or role, and click OK.

2. In the content tree, click the relevant item, and in the Security group, click Assign.

   Assi" height="457" width="676" alt="cms_all_SecurityEditor_AssignButton" title="The Security Editor - Assign button" class="documentimage" doc-fancy-image="">

3. In the Assign Security Rights dialog box, in the Permissions for section, grant or deny access to the item, the descendants, or both.

   For example, to deny the security account *Write* and *Rename* access rights to the *Our-Partners* item only, click the for the item access rights. This overrides the inherited access rights for the *Our-Partners* item and its descendants. To allow the *Write* and *Rename* access rights to the subitems of *Our-Partners*, click the for the Descendants access rights.

   Note

   In the Assign Security Rights dialog, you only need to grant or deny the access rights that you want to be different from the inherited access rights.

4. Verify your settings in the Access Viewer, where you can see the results of your settings on the descendants as well. In the Security Editor, in the Tools group, click Access Viewer.

   cms_all_AccessViewer_DenyParentItemWriteRename

   Note

   Remember that the user can be a member of multiple roles, which can have access rights that overrule the access rights that you have just assigned.

Assign access rights to multiple items

With the Sitecore search functionality, you can assign the same access rights to multiple items at the same time. For example, if you use item buckets that contain hidden items, you can use the search functionality to assign access rights and inheritance specifically to these hidden items.

Assign access rights to multiple items:

1. In the Content Editor or the Experience Editor, create a search and filter the results to list only the items that you want to assign the security rules to.

   cms_ce_SearchTab_SearchWithTemplates

2. Click the drop-down arrow to the left of the search field, click Search Operations, and then click Apply security rule.

   cms_ce_SearchTab_SearchOperations_ApplySecurityRule

3. In the Assign Security Rights dialog box, in the Role and users section, click the account that you want to assign the security rights to.

   Note

   If the appropriate account is not in the list, click Add to add the relevant account to the list.

   Per" height="466" width="348" alt="cms_all_ApplySecurityRights_MyRole" title="The Assign Security Rights dialog box" class="documentimage" doc-fancy-image="">

4. In the Permissions for section, click the icons or for the relevant security rights and then click OK.

   Important

   Even though the Permissions for section indicates that the access rights are assigned to the first item in the search result, they are in fact assigned to all the items included in the search result.

5. To verify your settings in the Access Viewer, in the Security Editor, in the Tools group, click Access Viewer.

   Note

   Remember that the user can be a member of multiple roles, which can have access rights that overrule the access rights that you have just assigned.

Send feedback about the documentation to docsite@sitecore.net.

Assigning access rights

In Sitecore, you can assign access rights to a security account to determine the access that a user has to the items and functionality in Sitecore. For example, the access rights on a security account can determine whether the user or role has the right to create items, delete items, or to push items through a workflow.

Each access right has one of three possible settings. They are:

- Allow – grants the associated access rights for the selected account.
- Deny – denies the associated access right for the selected account.

- Inherit – neither grants nor denies an access right. Instead, you can use this setting to allow or deny the item the right to inherit the access rights that are assigned to the parent item.

In Sitecore, when you assign access rights to items, they always inherit the access right that is assigned to their parent item in the content tree. However, if you need to, you can overrule the inherited rights on an item by assigning access rights specifically on the item or denying the item the right to inherit.

Assigning access rights to roles rather than users

You can assign access rights to both users and roles. However, if you make your users members of roles and assign the access rights to the roles instead of the user, you simplify maintenance. In this way, you can assign and revoke access rights to multiple users by assigning or removing memberships to roles instead of having to do this for each individual user account.

For example, if an employee leaves your company or moves to another department, you simply remove them from certain roles and make them members of other ones. Similarly, when you hire new employees, you can just make them a member of the roles that possess the relevant access rights.

By only assigning access rights to roles, you also make it easier to control a user's individual access rights when you have to. For example, if you want to ensure that a user has access to a particular item for a limited period, you do not have to study all the roles that the user belongs to, you just grant the relevant access rights to the user’s security account. This setting overrules the access rights specified for the roles that the user is a member of. To revert to the standard settings, you just remove the specified access rights from the user’s security account.

Conflicting access rights

A user can be a member of many different roles, and roles can also be members of other roles. When a role is a member of another role, the access rights of both roles are combined to give the users who are members of these roles the accumulated access rights of both roles.

If there are conflicting access rights between the user account and the roles, the following general rules apply:

De" height="201" width="624" alt="diagram_security" title="Security diagram" class="documentimage" doc-fancy-image="">

- Access rights – *Denied* overrides *Allowed*.
- Security accounts – Access rights assigned to a user account override access rights assigned to a role.
- Item – Access rights assigned specifically on an item override access rights specified for the descendants on the parent item.

  Access rights assigned specifically on an item or on the descendants of an item override the *Inheritance* access right.

Rule	Example
When an access right is not specified, it is *Denied*.	The default value for access rights is *Denied*. Therefore, if an access right is not specified for an item, the security account does not have access to the item.
If nothing is specified for the *Inheritance* access right, inheritance is *Allowed*.	The default value for the *Inheritance* access right is *Allowed*. Therefore, if *Inheritance* is not denied, the item inherits the access rights from its ancestors.
Denied overrules *Allowed*.	If a user is a member of two roles, one that explicitly grants them an access right to an item and one that explicitly denies them the same access right to the item, they are denied the access right. This also applies to the *Inheritance* access rights.
Access rights assigned to a user account overrule the access rights assigned to a role.	If an access right to an item is granted for a user account but denied for a role that the user account is a member of, then the user is granted the access right.
Access rights specifically assigned to an item for a user account overrule the access rights that are specifically assigned to an item for a role that the user is a member of.	If a user is a member of several roles and one of these roles is specifically denied an access right to an item, the user is denied the access right. However, if the user’s security account is specifically granted the same access right to the same item, the user is granted the access right.
Specifically assigned access rights to a user account overrule specifically assigned access rights to a role that the user is a member of.	If an access right for a user account is specifically granted to the descendants of an item and one of the roles that the user is a member of has the same access right specifically denied for the descendants of the item, the access right is granted to the descendent item.
Access rights specifically granted for an item, to either a user or a role, overrule the *Inheritance* access rights and any rights assigned to the descendants of the parent item.	If a user is a member of two roles, one that does not grant the user to inherit an access right to an item and another that explicitly grants the same access right, then the user is granted the access right.

Send feedback about the documentation to docsite@sitecore.net.

Restrict access to the client

When you configure servers for different purposes, depending on the role, you may need to disable the Sitecore client. For example, if you configure a content delivery server or processing server it is not necessary to access the client application, so in this case disabling the client is recommended.

To prevent unauthorized access to the Sitecore client interfaces, you need to restrict access on every instance of your content delivery or processing environments.

There are two ways to restrict access to the client:

- Implement IP-based security restrictions

- Disable Anonymous IIS access

Implement IP-based security restrictions

To restrict access to client interfaces, you can implement IP-based security restrictions.

Note

The steps you follow to implement IP-based security restrictions vary depending on which operating system you have.

To implement IP-based security restrictions in Windows 7:

1. Open the Control Panel.
2. In the Control Panel, click Programs.
3. Under Programs and Features, click Turn Windows Features on or off.
4. In the Windows Features window, select IP Security.

Picture 5

For instructions on how to configure IP-based security restrictions in IIS 7 and later, see http://www.iis.net/ConfigReference/system.webServer/security/ipSecurity.

For instructions on how to configure IP-based security restrictions in IIS 6 and earlier, see http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/128d26dd-decb-42f9-8efb-30724d1a2f29.mspx?mfr=true.

Disable Anonymous IIS access in versions earlier than Sitecore 8.2, Update 3

Another way to restrict access to the client is by disabling Anonymous IIS access to the following folders and files in your *Website\sitecore* folder:

- `admin folder`
- `login folder`
- `shell folder`
- `default.aspx page`

Note

You need to exclude the */sitecore/service* folder from the IIS restrictions because it contains several service .ASPX pages that are used when reporting conditions or information back to the web client. For example: *404 Page Not Found* and *403 Forbidden*

You can move files from the */sitecore/service* folder to sit outside the */sitecore* folder, but you must also remember to update the following settings in the `web.config` file: `ErrorPage`, `NoAccessUrl`, `NoLicenseUrl`, `LayoutNotFoundUrl`, `ItemNotFoundUrl`, `LinkItemNotFoundUrl`

To disable Anonymous IIS access:

1. Open Internet Information Services (IIS).
2. In IIS, click Features View and then in the Security category, select Authentication.
3. In your website folder structure, click, for example, the *admin* folder.

Picture 7

4. To set *Anonymous Authentication* to *Disabled*, in the Actions panel, click Disable.

Disable Anonymous IIS access in versions later than Sitecore 8.2, Update 3

In Sitecore 8.2, Update 3 and later, Forms authentication is the default setting of authentication in the `web.config` file. This means that the procedure described above does not work. You can work around this issue in two ways:

- Disable Forms authentication
- Restrict access with .Net Authorization

Disable Forms authentication

Disable Forms authentication this way:

- In the `web.config` file, change `Forms` to `None` in the authentication node:

  `<authentication mode="None">`

This has an effect for users who need to access the Sitecore client page (such as the "shell" or other pages available after login). These users will meet an access error (error 401).

It is safe to disable Forms authentication for a user (such as the Content Delivery client) that does not need to login or access the client page.

Restrict access with .NET Authorization

You set up .NET Authorization in the Internet Information Services (IIS) manager.

You must deny anonymous access to the following folders and files in your *Website\sitecore* folder (*sitecore* in Sitecore 9.0 and later):

- `admin folder`
- `login folder`
- `shell folder`
- `default.aspx page`

To deny access in the IIS Manager:

1. Select each of the items in the list above (three folders and one file) in the IIS Manager.
2. For each item, click the .NET Authorization Rules:

   Picture 1

3. Click Add Deny Rule… in the Actions panel:

   Picture 2

4. Select All anonymous users and OK:

   Picture 3

This creates a directory specific `web.config` file for each item in the list. [How To Make Application and Directory-Specific Configuration Settings in an ASP.NET Application] at the Microsoft support site describes how to add these file locations to the application-wide `web.config` file with nodes similar to this: `<location path="sitecore/shell">`.

Send feedback about the documentation to docsite@sitecore.net.

The access rights

You can assign access rights to an account on an item level. The following access rights can be granted or denied to individual users or roles, or they can be inherited from the parent item.

Note

To view more access rights in the Security Editor, in the Security group, click Columns. In the Columns dialog box, select the access rights that you want to display in the Security Editor and click OK.

Access right	Description
Field Read	Controls whether a user can view a specific field on an item.
Field Write	Controls whether a user can edit a specific field on an item.
Read	Controls whether a user can see an item in the content tree and/or on the published website, including all the properties and field values.
Write	Controls whether a user can edit field values. The *Write* access right requires the *Read* access right and *Field read* and *Field write* access rights for individual fields (*Field read* and *Field write* are allowed by default).
Rename	Controls whether a user can change the name of an item. The *Rename* access right requires the *Read* access right.
Create	Controls whether a user can create child items. The *Create* access right requires the *Read* access right.
Delete	Controls whether a user can delete an item. The *Delete* access right requires the *Read* access right. Important This command also deletes all child items, even if the *Delete* access right has been denied for the account for one or more of the subitems.
Administer	Controls whether a user can configure the access rights of an item. The *Administer* access right requires *Read* and *Write* access rights.
Language Read	Controls whether a user can view a specific language version of an item in the Sitecore Clients. Does not influence the web site.
Language Write	Controls whether a user can edit a specific language version of an item in the Sitecore Clients. Does not influence the web site.
Show in Insert	Controls whether a template is shown in the Content Editor in the Insert Options list and in the Experience Editor in the Insert dialog box.
Workflow State Delete	Controls whether a user can delete items when they are in a specific workflow state.
Workflow State Write	Controls whether a user can update items when they are in a specific workflow state.
Workflow Command Execute	Controls whether a user can execute a specific workflow command.
Customize Profile Key Values	Controls whether a user can customize the profile key values on a profile card.
Create Bucket	Controls whether a user can create an item bucket.
Revert Bucket	Controls whether a user can revert an item bucket to a regular item.
Field Remote Read	Controls whether the Item Web API services can access (read, retrieve) the fields of an item. If you want a field to be available for requests, you should allow this access right for the field. Note This access right is only applicable on fields and by default set to *Denied*.

Inheritance	Controls whether security rights can be passed from a parent item to the child items. The security model supports the possibility to grant or deny the *Inheritance* access right on a per account basis (it applies to all access rights). The inheritance settings that you choose, only apply to the selected account.

Send feedback about the documentation to docsite@sitecore.net.

The Inheritance access right

The *Inheritance* access right is a setting that determines whether an item can inherit its ancestors' access rights for a specific security account.

You can use the *Inheritance* access right to streamline the process of assigning access rights. In certain situations, this can spare security administrators the tedious task of assigning each role explicit access rights to every item in the content tree.

By default, an item inherits the access rights that are assigned to the items that are higher up the content tree (the ancestors). For example, a security administrator can configure the security settings of a single item and all the items that are lower down the content tree (the descendants) automatically inherit the same settings. However, if there is an item that should not inherit any security settings from its ancestors, you can use the *Inheritance* access right to deny the item the right to inherit the security settings.

Important

Because items by default inherit the access rights from their ancestors, you never have to actually grant an item the *Inheritance* access right.

This topic covers:

Conflicting access rights

If there are conflicting inheritance or descendant access rights assigned to a user or the roles that a user is a member of, the following rules apply:

Rule	Example
An *Inheritance* access right that is denied to a role or a user overrules the *Inheritance* access rights specified on other roles or users.	If a user is a member of two roles, one that allows the user to inherit an access right to an item and another that does not allow the user to inherit the same access right to the item, then the user is denied the access right.
Access rights explicitly granted for an item, on either a user or a role, overrule the *Inheritance* access rights and any rights assigned to the descendants of the parent item.	If a user is a member of two roles, one that does not allow the user to inherit an access right to an item and another that explicitly grants the same access right, then the user is granted the access right.
Explicitly assigned access rights on a user account overrule explicitly assigned access rights on a role that the user is a member of.	If an access right on a user account is explicitly granted to the descendants of an item and one of the roles that the user is a member of has the same access right explicitly denied for the descendants of the item, the access right is granted to the descendent item.

Examples of how to control access with the Inheritance access right

The following examples show you how to control the access rights to items and their descendants with the *Inheritance* access right.

Deny an account all access rights to an item and its descendants

This example shows how you can deny an account the access right to an item and all its descendants.

For example, you can grant the *My Role* security account the *Write, Rename, Create, and Delete* access rights to the *About Us* item and then deny the *Our-Partners* subitem the right to inherit these access rights.

To do this in the Security Editor, you deny the *Our-Partners* item the *Inheritance* access right. This denies the selected account every access right to the item and its descendants, including the *Read* access.

cms_all_SecurityEditor_Inheritance

In the Access Viewer, you can view how the settings in the Security Editor affect the access rights for the descendants of the *About-Us* item.

In this example, the *My Role* account no longer has any access to the *Our-Partners* item or its subitems and because the *Press-Lounge* item is still allowed to inherit the access rights of its ancestors, the *My Role* account still has the access rights to that item.

To see the details of how access rights have been granted or denied for a specific item, you can click the access right and the details appear in the pane on the right of the Access Viewer. In this example, the *Write* access for the *Our-Partners* item has been denied the right to inherit the settings from the *About-Us* item.

cms_all_AccessViewer_DenyInheritance

Note

The *Inheritance* access right will not deny the user access to the item in question if the user is a member of another role that grants them access to the item. In addition, the access rights that are explicitly assigned on an item, overrule the inheritance settings.

Grant an account access rights to an item and deny it to the descendants

This example describes how you can grant access rights to an item but deny them to the item's descendants. For example, you can grant the *My Role* account all access rights to the *Our-Partners* item but deny everything but the *Read* access right to the *Inventory-Partners* and *Construction-Partners* descendants.

To do this, you must assign the access rights explicitly to the *Our-Partners* item in the Assign Security Rights dialog box:

- In the Permissions for section, allow descendants to inherit the *Read* access rights. This setting overrules the settings in the Inheritance section.
- In the Inheritance section, deny only the descendants (and not the item) to inherit the access rights from the parent item.

cms_all_AssignSecurityRightsDialog_DenyDescendants

In the Security Editor, the two icons for the access rights of the *Our-Partners* item indicate that you have specified different access rights and inheritance settings for the item and its descendants. The upper icon displays the access rights for the item and the lower icon displays the access rights for its descendants. In this example, the *My Role* account has full access rights to the *Our-Partners* item but only *Read* access rights to its descendants.

cms_all_SecurityEditor_ItemAndDescendantsRights

In the Access Viewer, you can click an access right to view the details of the settings in the pane on the right:

cms_all_AccessViewer_DenyDescendants

Deny an account access rights to an item and grant them to its descendants

This example describes how you can use the *Inheritance* access right to ensure that a security account has access rights to the descendants of an item but not to the item itself. For example, you can deny the *My Role* account access rights to the *Our-Partners* item but allow the account full access to its *Inventory-Partners* and *Construction-Partners* descendants.

To do this, you assign the access rights explicitly to the *Our-Partners* item in the Assign Security Rights dialog box:

- In the Permissions for section, grant the *Read* access rights to the item. This setting overrules the settings in the Inheritance section.
- In the Inheritance section, deny only the item the right to inherit the access rights (and not the descendants).

cms_all_AssignSecurityRightsDialog_DenyParentItem

In the Access Viewer, you can view the details of the settings. The *My Role* account does not have full access to the *Our-Partners* item but does have full access to its two descendants. This is achieved efficiently by only using the *Inheritance* access right and not by denying and granting access rights on each item.

cms_all_AccessViewer_DenyParentItem

Send feedback about the documentation to docsite@sitecore.net.

View the access rights of a security account

In the Security Editor, you can see the access rights that are assigned to an account, but from the Access Viewer and from the Content Editor, you can see the full result of the combinations of access rights for an individual item.

This topic outlines how to:

- View the access rights in the Access Viewer
- View and edit the access rights in the Content Editor

View the access rights in the Access Viewer

To see the access rights that each role has to items in the Access Viewer:

1. Log in to Sitecore, on the Launchpad, click Access Viewer.

cms_LaunchPad_AccessViewer

2. In the Access Viewer, in the Roles and Users group, click the security account that you want to see the assigned access rights for.
   - If the role or user is not visible, click the drop-down arrow and click the security account from the list.
   - If the role or user is not visible in the list, click Account and in the Add an Account dialog box, in the Account Type section, click the relevant account type and then the relevant role or user from the list.

   cms_all_AccessViewer_RolesAndUsersField

3. For each item in the content tree, you can see the access rights that the selected role or user has assigned.

   Important

   The content tree in the Access Viewer does not update automatically. You must collapse and expand the relevant nodes to refresh them and see the latest changes.

4. To see the details for an individual access right, click the relevant access right and the result of all the access rights that are assigned to an item appear in the pane on the right.

   In this example, you can see that *My Role* has been granted the *Write* access right to *My item*, but *My item* is part of the *Sample Workflow*, which denies the access rights *Write* and *Delete*. Therefore, the account *My Role* denied those two access rights for that particular item. This information is only available to you from the Access Viewer.

   The keys in the Access Viewer:

   - Not Specified – if nothing is specified on an access right, it is denied by default.
   - Allowed – the security account is granted the access right on the item.
   - Denied – the security account is denied the access right on the item.
   - Item vs. Descendant Right – the access right assigned to the item is different from the access right assigned to the descendants. The upper icon applies to the item and the lower icon applies to the descendants.
   - Protected – a protected item is unavailable in the content tree and therefore protected from any editing. You cannot assign any access rights to a protected item.
   - Not Applicable – no access rights have been specified for the item. This key appears only in the item details in the pane on the right.

View and edit the access rights in the Content Editor

To see an overview of the security accounts that have been assigned explicit access rights to an item:

1. Open the Content Editor and in the content tree, click the relevant item.
2. Click the Security tab, and in the Security group, click Details.

   The Security Details tab in the content pane displays a list of the roles that have been assigned explicit access rights to the current item.

cms_ce_SecurityTab_SecurityDetailsContenttab

3. If there are no access rights assigned explicitly to the item, click the link *apply access rights* to assign the relevant access rights.

cms_ce_SecurityTab_SecurityDetails_NoAccessRightsAssigned

4. To change the access rights assigned explicitly for the selected item, in the Security group, click Assign to open the Assign Security Rights dialog box.

cms_ce_SecurityTab_SecurityGroup_AssignSecurityRights
Dialog

Send feedback about the documentation to docsite@sitecore.net.

Change your Sitecore password

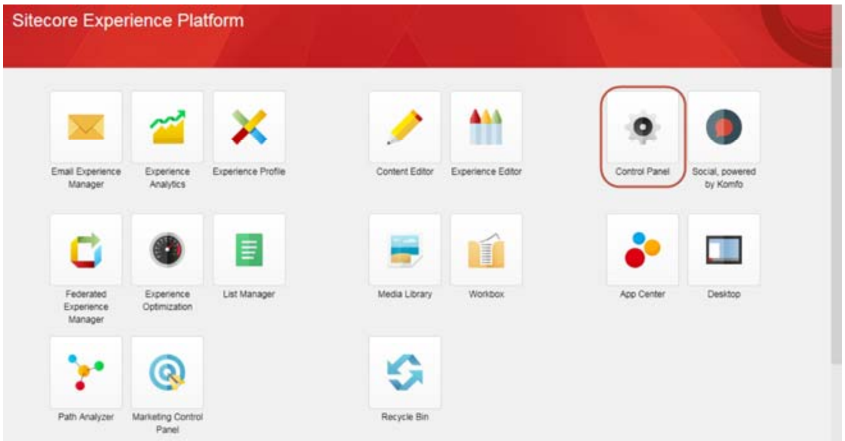
When you log in to Sitecore for the first time, one of the first things you must do is to change your password to one that you know and can remember. You cannot change your user name.

Note

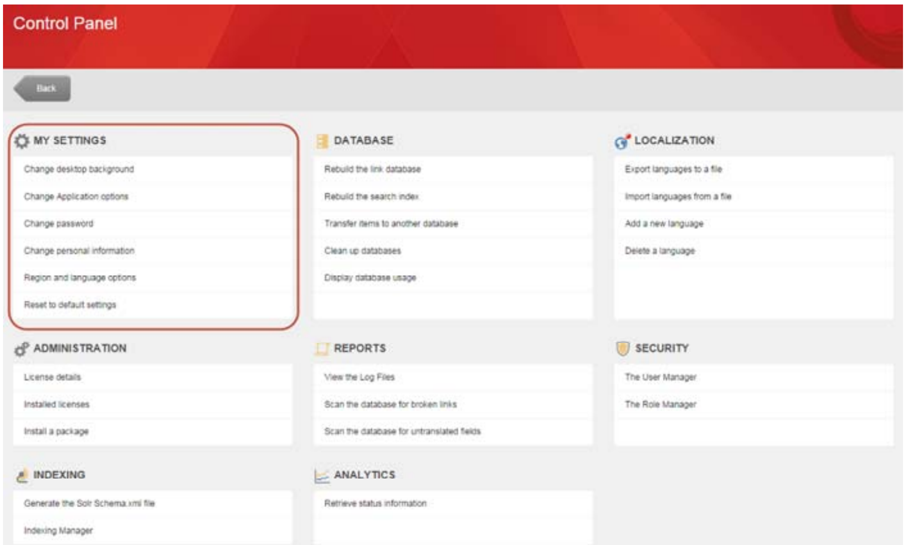
If you have forgotten your password, on the login page, you can submit a request to have your current password sent to you in an email. However, if you have entered the wrong password a number of times and your account has been locked, the email will not be sent to you. In this case, you must ask your system administrator to unlock your account and create a new password for you.

To change your Sitecore password:

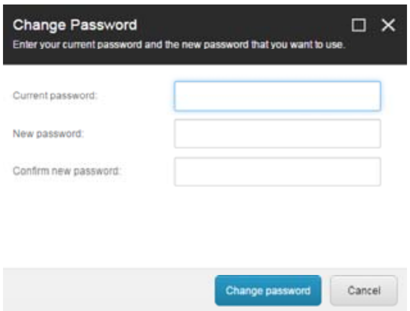
1. Log in to Sitecore, and on the Sitecore Launchpad, click Control Panel.



2. In the Control Panel, in the My settings section, click Change password.



3. In the Change Password dialog box, enter your current password and then enter and confirm your new password.



4. Click Change password.

Send feedback about the documentation to docsite@sitecore.net.

Configure the password policy

When you setup Sitecore security, you should also consider how you want to set up the password policy, such as the minimum length and strength of your users' passwords and if you want to allow users who forget their password to request a new password in an email message.

This topic describes how to

- [Specify your password policy]
- [Enable the forgotten password functionality]

Specify your password policy

The security architect can specify the password policy to be enforced on your website. The parameters that can be specified include the length and strength of the passwords that users must use, as well as the number of times that a user can enter an incorrect password before they are locked out.

To specify the password policy:

1. In Windows Explorer, browse to the folder where the website is stored, typically, `C:\Inetpub\wwwroot\SitecoreWebsite\WebSite`.
2. Open the `Web.config` file in Notepad and scroll down to the following section:

```
<membership defaultProvider="sitecore">
  <providers>
    <clear />
    <add name="sitecore" type="Sitecore.Security.SitecoreMembershipProvider, Sitecore.Kernel" realProviderName="sql" providerWildcard="%" raiseEvents="true" />
    <add name="sql" type="System.Web.Security.SqlMembershipProvider" connectionStringName="core" applicationName="sitecore" minRequiredPasswordLength="1"
    minRequiredNonalphanumericCharacters="0" requiresQuestionAndAnswer="false" requiresUniqueEmail="false" maxInvalidPasswordAttempts="1" />
    <add name="switcher" type="Sitecore.Security.SwitchingMembershipProvider, Sitecore.Kernel" applicationName="sitecore" mappings="switchingProviders/membership" />
  </providers>
</membership>
<roleManager defaultProvider="sitecore" enabled="true">
  <providers>
```

3. Edit the following properties:

Property	Defines
minRequiredPasswordLength	The minimum number of characters that a password must contain.
minRequiredNonalphanumericCharacters	The minimum number of non-alphanumeric characters that a password must contain. Non-alphanumeric characters are any characters that do not contain the value of a number or a letter, for example, !@#$%&*() Default value = 0.
maxInvalidPasswordAttempts	The maximum number of times that a user can enter an incorrect password before their security account is locked out.

Enable the forgotten password functionality

You must also edit the `Web.config` file to enable Sitecore to send an email message to users who use the *Forgot Your Password* functionality and request to receive a new password in an email message.

To enable the *Forgot Your Password* functionality:

1. Open the `Web.config` file in Notepad.
2. Scroll down to the *Mail Server* section:

```
<!--  MAIL SERVER

      SMTP server used for sending mails by the Sitecore server

      Is used by MainUtil.SendMail()

-->

<setting name="MailServer" value="mail.server.net" />

<!--  MAIL SERVER USER

      If the SMTP server requires login, enter the user name in this setting

-->

<setting name="MailServerUserName" value="" />

<!--  MAIL SERVER PASSWORD

      If the SMTP server requires login, enter the password in this setting

-->

<setting name="MailServerPassword" value="" />

<!--  MAIL SERVER PORT

      If the SMTP server requires a custom port number, enter the value in this setting.

      The default value is: 25

-->

<setting name="MailServerPort" value="25" />
```

3. Enter the address of your mail server in the `<setting name="MailServer" value="mail.server.net" />` section.

   If necessary, you can also specify the user name, password, and the server port.

4. Save the `Web.config` file.

You need to change the sender of the *Forgot your Password* email to a valid email address. You can also the subject and content of the email. To do this, follow these steps:

1. Log in to Sitecore as an administrator.
2. In the Launchpad, open the Desktop, and select the Core database.
3. In the Content Editor, navigate to */sitecore/system/Settings/Security/Password recovery/Password Recovery Email*.

4. Enter a valid email address in the Sender email address field, and change the other fields as you need..

Note

You must configure your SMTP server to allow emails to be sent from the email address you specified as the Sender email address.

Send feedback about the documentation to docsite@sitecore.net.

Kick off or add a user

When you log into Sitecore, if the maximum number of allowed users has been reached, you can either force the logout of a specific user or increase the number of allowed users. This ensures that you can always log in to Sitecore if necessary. To do this you need to have administrator rights.

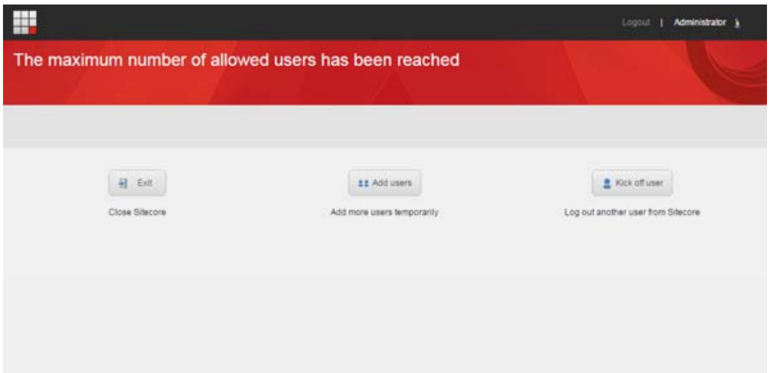
This topic outlines how to:

- Kick off a user from Sitecore
- Increase the number of allowed users

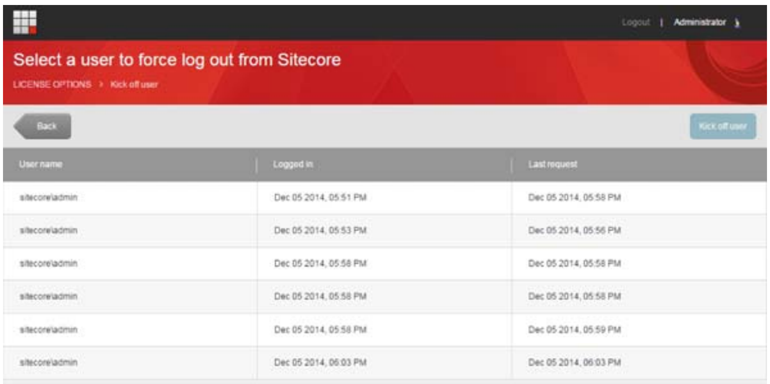
Kick off a user from Sitecore

To force the logout of a user from Sitecore when you have reached the maximum number of allowed users:

1. On the page that opens when you log in and the maximum number of allowed users has been reached, click Kick off user.



2. On the page Select a user to force log out from Sitecore, in the table, click the user that you want to kick off from Sitecore and click Kick off user. The user is automatically logged out from Sitecore.



3. The Launchpad opens and you are logged in to Sitecore.

Note

A user that is logged in two different browsers and therefore has two sessions will appear twice in this list.

When a user successfully logs in, a cookie (.ASPXAUTH) is set. As long as this cookie exists, the user remains logged in. The Authentication.ClientSessionTimeout setting in the Sitecore.config file (Web.config in Sitecore 8.) specifies the timeout value of this cookie. The default setting is 60 minutes. You can change this value if you want user sessions to time out faster.

Increase the number of allowed users

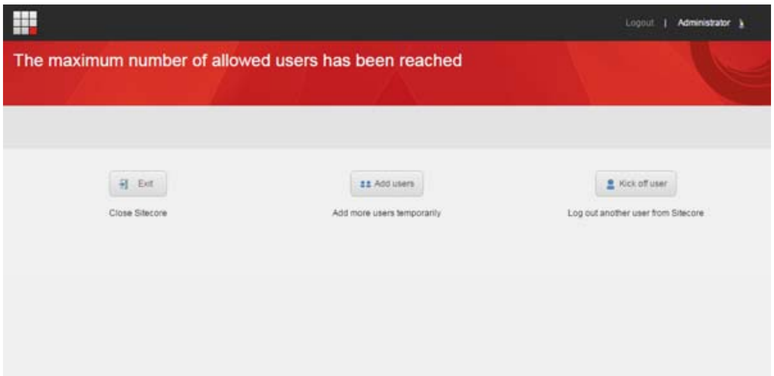
You can increase the number of allowed users up to 24 hours even if your license does not include these additional users.

Important

Your license agreement regulates the maximum user count. If you use this functionality too often, this can be a breach of your license agreement and Sitecore can charge you for the extra users.

To increase the number of allowed users in your Sitecore installation:

1. On the page that opens when you log in and the maximum number of allowed users has been reached, click Add users.



2. On the Boost Sitecore page, enter the number of extra users that you want to have access to Sitecore, and click Boost.

3. Confirm your request and click Finish.
4. The Launchpad opens and you are logged in to Sitecore.

Send feedback about the documentation to docsite@sitecore.net.

Log in to the Sitecore Experience Platform

To work on your website, you must log in to Sitecore and select the application you want to work in.

Log in to the Sitecore Experience Platform

To log in to the Sitecore Experience Platform:

1. In a browser, enter the name of your website followed by */sitecore* in the address field. For example, http://www.mywebsite.net/sitecore.

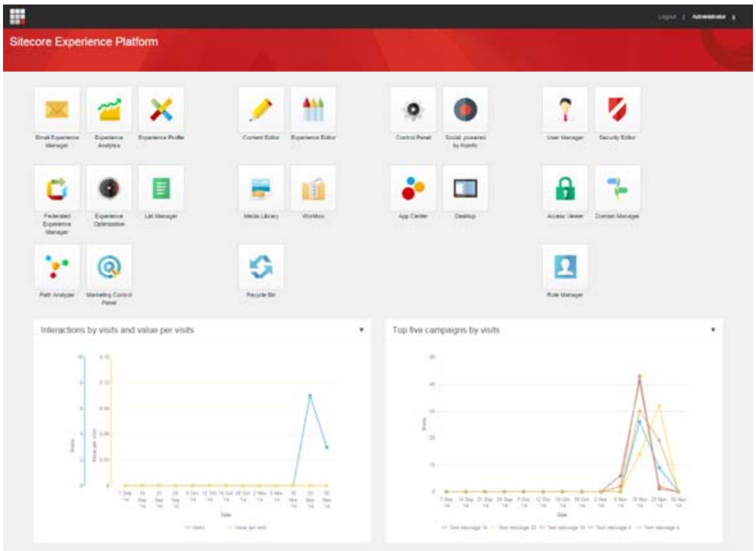
2. On the login page, enter your user name and password, and click Log in.

   Note

   The first time you log in to Sitecore, you should change your password in the Control Panel to one that you know and can remember.

3. On the Launchpad, you see an overview of the applications that are available to you and the two reports:
   - Interactions by visits and value per visit
   - Top five campaigns by visit

   To change the view of the reports, click the down arrow for a report and click another view.



4. Click the application that you want to work in.

   Note

   The applications that are available to you on the Launchpad vary depending on your role and access rights.

Send feedback about the documentation to docsite@sitecore.net.

Add a user to a role

When a Sitecore user's area of responsibility changes, the roles that the user is a member of must also reflect that change. To do this, you must make the user a member of a different and more relevant role or roles and remove the user from the role that they should no longer be a member of.

To add or remove a user from a role:

1. Open the User Manager, click the relevant user, and then click Edit.
2. To add or remove a user from a role, in the Edit User dialog box, on the Member Of tab, click Edit.

   cms_all_EditUserDialog

3. In the Edit User Roles dialog box, in the Available Roles section, select the roles that you want to make the user a member of and then click Add.
   - You can press SHIFT or CTRL to select several roles.
   - You can double click a role to add or remove it.

cms_all_EditUserRolesDialog

If the relevant roles are not displayed on the first page, use the navigate buttons to browse through all the roles.

4. To revoke a user's membership of a role, in the Selected Roles section, click the relevant role and then click Remove.
5. When all the relevant roles are selected, click OK.

Send feedback about the documentation to docsite@sitecore.net.

Create a user

In Sitecore, you use the User Manager to create users and manage the roles that they are members of.

To create a user in the User Manager:

1. Log in to Sitecore and, on the Launchpad, click User Manager.

cms_LaunchPad_UserManager

2. In the User Manager dialog box, in the Users group, click New.

cms_all_UserManagerDialog

3. In the Create a New User dialog box, enter the relevant information about the new user. For example, enter the user name and password that the user must use for authentication when they log in to Sitecore.

Important

Important

Passwords are case-sensitive in Sitecore but user names are not.

The Create a New User dialog box contains the following fields:

Field	Value
User name	Enter the name of the user in Sitecore. (mandatory)
Domain	Enter the domain that the user has access to.
Full name	Enter the full name of the user.
Email	Enter the user’s email address. (mandatory)
Comment	Enter any relevant comments.
Password	Enter the password of the new user. (mandatory) Users can change it after they log in to Sitecore.
Confirm password	Confirm the password (mandatory).
Roles	Enter the roles that you want to make the user a member of. Click Edit to add the user to one or more roles.
User profile	Enter the type of user you are creating.

4. Click Next to validate the information that you have entered and then click Finish.

When you have created the user, you must add the user to one or more roles. These roles determine the user's access rights and thereby which items that the user can access in Sitecore and the actions that the user can perform on these items.

Send feedback about the documentation to docsite@sitecore.net.

Create and edit a security domain

You can create extra domains, for example, for the website of another company or a subsidiary. This is a task for a security architect.

This topic outlines how to:

- Create a domain
- Assign security accounts to a domain
- Edit a domain
- Delete a domain

Create a domain

To create a domain:

1. Log in to Sitecore, and on the Launchpad, click Domain Manager.

Doma" height="315" width="666" alt="cms_LaunchPad_DomainManager" title="The Launchpad - Domain Manager" class="documentimage" doc-fancy-image="">

2. In the Domain Manager, in the Domains group, click New.

cms_all_DomainManagerWindow

3. In the New Domain dialog box, enter the name of the domain and, if you want the domain to be managed locally, select the Locally Managed Domain check box.
4. Click OK. This adds the new domain to the `/App_Config/Security/Domains.config` file. For details on how to configure the new domain, see the comments in that file.

   Note

   In a multi-server setup, you must manually add the new security domain to the *Domains.config* file on each server.

   Sitecore will add the anonymous user to a new domain only when you restart Sitecore.

Assign security accounts to a domain

Because a domain is also a security construct, it must contain users and roles before it has any meaning. You can only assign security accounts to a domain when you create a new user in the User Manager or create a new role in the Role Manager. You cannot change the domain of an existing account.

Note

A security account can only be assigned to one domain. If, for example, a user needs to access multiple domains, you must create separate roles for each domain that they need to access and make the user a member of all the relevant roles.

Edit a domain

When you edit a domain, the only setting you can change is whether the domain should be a locally managed domain.

Note

In a locally managed domain, the users and roles are domain specific and the users can only see the items in the domain that they belong to and not the other domains in the system.

To edit a domain:

1. Open the Domain Manager and click the domain that you want to edit.
2. In the Domains group, click Edit.
3. In the Edit Domain dialog box, select or clear the Locally Managed Domain check box.

Delete a domain

If you no longer need a domain, you can delete it.

Important

When you delete a domain, the security accounts that belong to the domain are not deleted. To make sure that you do not have accounts that cannot be used, you should either delete the users or roles or make sure that they are members of another role that belongs to a domain.

To delete a domain:

1. Open the Domain Manager and click the domain you want to delete.
2. In the Domains group, click Delete.

Send feedback about the documentation to docsite@sitecore.net.

Create and set up a role

You can use roles to manage website authorization. Roles allow the grouping of users into structured units, such as managers, sales staff, anonymous users, and so on. This makes it easier to organize security access because you can use a single role to assign security access rights to multiple users. Roles give you the flexibility to change permissions and to add or remove users without having to make changes to the whole website.

Users can belong to more than one role, giving them different access rights to different areas of a site. Users who are members of multiple roles gain their access rights from all the assigned roles.

In Sitecore, you use the Role Manager to create new and manage existing roles.

This topic outlines how to:

- Create a role
- Assign access rights to a role
- Add an account as a member of a role
- Delete a role

Create a role

To create a role:

1. Log in to Sitecore, and on the Launchpad, click the Role Manager icon.

   cms_LaunchPad_RoleManager

2. In the Role Manager window, on the ribbon, click New.

   Role dial" height="429" width="693" alt="cms_all_RoleManagerWindow" title="The Role Manager" class="documentimage" doc-fancy-image="">

3. In the New Role dialog box, enter the name and domain of the new role.

   Important

   To ensure that all your roles are available to all of the domains in your system, you should keep all your roles in the default Sitecore domain. Once you create a role and make it specific to a domain, you cannot change the domain that it belongs to.

4. Click OK.

Assign access rights to a role

By default, the new role has read access to everything. To finish setting up the role, you assign the relevant access rights.

Add an account as a member of a role

When you have created a new role, you can add members to the new role or you can make the new role a member of another role. In Sitecore, you can make any security account (users or roles) a member of a role. To make a security account a member of a role:

1. In the Role Manager, click the role that you want to add a security account to and then click Members.

cms_all_RoleManager_MembersButton

Note

To make the selected role a member of another role, click Member Of, and click the roles that you want this role to be a member of.

2. In the Members dialog box, you can see a list of all the security accounts that are members of the role. To add a new security account to the list, click Add.

   cms_all_RoleManager_MembersDialog

   Note

   To delete a security account from a role, click the account that you want to delete and click Remove.

3. In the Add an Account dialog box, in the Account Type section, click the type of account (user or role) that you want to add.

   cms_all_RoleManager_AddAnAccountDialog

4. Click the user or role that you want to make a member of the role and then click OK.

   If there is more than one page of users or roles, use the buttons at the bottom of the window to browse through the list of users or roles.

Delete a role

As a security administrator, you often remove roles and users from the security system as your company changes and grows.

When you delete a role, Sitecore:

- Removes membership of the role from all the security accounts that were members of the role.
- Removes all the access rights associated with this role from all the users who were members of the role.
- Removes the account definitions.
- Does not remove the access rights associated with the role. The access rights are still stored on the individual items in the content tree.

  This means that if you create a new role with the same name as an old one, the new role is granted the same access rights as the old one. The new role does not have any members.

To delete a role:

1. Open the Role Manager and click the role that you want to delete.
2. In the ribbon, click Delete.

Send feedback about the documentation to docsite@sitecore.net.

Edit and manage a user account

As a security administrator, you can edit and manage many aspects of a user account.

This topic outlines how to:

- Edit a user account
- Change a user's password
- Unlock a user’s security account
- Disable a user
- Delete a user

Edit a user account

In the User Manager, you can edit a users' account information, change the roles that they are members of, or delete them from Sitecore.

To edit a user:

1. In the User Manager, click the relevant user and in the Users group, click Edit.
2. Edit the user information on the following tabs:
   - The General tab – change the user's name and email address. You can also edit the image that is used as a portrait of the user in Sitecore.
   - The Member Of tab – edit the roles that the user is a member of and associate one or more domains with the user.

     Note

     If you want to allow users of a locally managed domain to see and manage more domains than the one they belong to, you can click Domains and associate the relevant domains to the user. Users of global domains can, by default, work with all domains, but if you associate one or more domains to this type of user, they are restricted to only seeing the domains you associated.
   - The Profile tab – in the User Profile section, specify the start URL that should open for the user when the user logs in. No matter what you choose, the user has access to all the same things.

Start URL	Description
Default	The Launchpad opens from where the user can open and view everything that the user has to access.
Content Editor	The Content Editor opens.
Experience Editor	The Experience Editor opens.
Preview	Sitecore opens in preview mode.
Desktop	Sitecore Desktop opens.
URL	Sitecore opens the custom URL that you enter in the field. For example, if you want Sitecore to open the Media Library on login.

   - In the Additional Properties section, edit information about the user profile that was selected for the user when their account was created.
   - The Language Settings tab – in the Sitecore Client section, specify the language and the regional code that the Sitecore client should use when the user logs in. In the Content section, specify the default language of the website content for the user.
   - The Information tab – view the history of the user. This includes when the user was created, the last login time, and so on.

cms_all_UserManager_EditUserDialog

3. When you have finished editing the user, click OK.

Change a user's password

If necessary, a security administrator can change a user’s password.

To change a user's password:

1. Open the User Manager and click the relevant user.
2. In the Users group, click Change password.
3. In the Change Password dialog box, in the Old Password field, enter the old password, and then enter and confirm the new password.

   cms_all_ChangePasswordGenerateDialog

   However, if you do not know the old password, you can randomly generate a new password.

4. To change the user’s password, click Generate.

   Note

   The user’s current password becomes invalid as soon as you generate the new password.

5. Copy the generated password to the clipboard and send it to the relevant user.

The user can then log in to Sitecore and change their password to one that they can remember. However, if the user is locked out of Sitecore, the user will not be able to log in (even with the new password) to change the password. In this situation, the security administrator must also unlock their security account.

Unlock a user’s security account

If a user is locked out of Sitecore, they must ask the security administrator to unlock their security account.

To unlock a user account:

1. Open the User Manager and click the relevant user. In the Locked column, an indicator shows if a user has been locked out of Sitecore.
2. In the Users group, click Unlock.

Disable a user

A security administrator will occasionally have to prevent users from accessing the system for a certain period, for example, if a user is on extended leave. In this situation, you can disable the user.

To disable a user:

1. Open the User Manager and click the relevant user. In the Locked column, *Disabled* indicates if a user has been disabled.

cms_all_UserManager_UserDisabled

2. To disable a user, in the Users group, click Disable. To enable the user again, click Enable.

Delete a user

As a security administrator, you must remove users and roles from the security system as your company changes and grows.

When you delete a user, Sitecore:

- Removes this user from all the roles that they are a member of.
- Removes the account definition item.
- Does not remove the access rights associated with the user. The access rights are still stored on the individual items in the content tree.

  This means that if you create a new user with the same name as an old one, the new user is granted the same access rights as the old one. However, the new user account does not automatically become a member of any roles.

  Note

  This is one of the reasons that you should only assign access rights to roles. In this way, you minimize the risk of inadvertently granting them individual access rights to items in the content tree. You can concentrate on managing the access rights of the roles that they are members of.

To delete a user:

1. Open the User Manager and click the user that you want to delete.
2. In the Users group, click Delete.

Send feedback about the documentation to docsite@sitecore.net.

Security domains

You can use security domains to manage user's access to different parts of Sitecore, for example, if you have multiple websites within a single system.

A Sitecore domain is a collection of security accounts (users and roles) that you can administer as a unit with common rules and procedures. A domain is used to collect security accounts that have some logical relationship, for example, all the accounts that have access to use the Sitecore clients could be stored in the Sitecore domain, whereas all the accounts with access to the published website could be stored in the Extranet domain.

Default domains

Sitecore contains the following default domains:

- Extranet — this domain is a website security domain that contains the user accounts that correspond to the visitors to the website. It also contains the customized roles that manage read access to the content of the website.

  Note

  Members of the Extranet domain that are also members of relevant Sitecore roles (for example, Sitecore Client Authoring) can access the Sitecore domain and use the client tools to edit the content of the website.

- Sitecore — this domain is an internal security domain that contains all the users who can access the Sitecore clients and the Sitecore Client roles that influence the functionalities that are available to users. It also contains the customized roles that control the access that users have to items.

  Note

  Members of the Sitecore domain can access the Sitecore client tools and edit the website if they have the appropriate access rights. Furthermore, they may be able to access the Extranet domain depending on how the developers and the security architect have designed the domain and the login page.

- Default – this is a virtual domain that only exists in memory. Sitecore is setup with Extranet as the default domain and most user accounts will be extranet\anonymous when they visit the website. However, if the website does not specify a default domain, then the users are set to default\anonymous.

Note

All these domains contain an *Anonymous* user and an *Everyone* role. The *Everyone* role automatically includes all users and the *Anonymous* user in the domain.

Generally, users will be members of roles in their domain and domains will not contain users from other domains. However, users can be members of roles in multiple domains, and roles can contain users from multiple domains.

Domain types

A domain can be of one of the following two domain types:

- Global domains -– users are able to access all the domains in the system. However, this depends on how the security architect has configured the system.

- Locally managed domains – users can only access a specific domain and not the other domains in the system. A local administrator, who cannot access other domains within the system, usually maintains a locally managed domain. Furthermore, users that belong to a locally managed domain will only see security accounts that belong in the same domain.

Send feedback about the documentation to docsite@sitecore.net.

The security roles

Sitecore comes with a series of predefined roles that you can use to manage user authorization on items and functionality.

Security role	Description
Everyone	All users are assigned the *Everyone* role. This role is a virtual role. It mirrors the Windows Everyone group. It does not exist in the role database but is only used for assigning and resolving security. You can use the *Everyone* role to assign access rights to every user or every user in a specific domain. The *Everyone* role is available as both a global role and a local role in every domain.
Analytics Advanced Testing	Gives the user access to see additional tabs and controls in the Marketing Control Panel. You typically give this role to optimization experts, who need expanded rights when performing tests, traffic allocation, and so on.
Analytics Content Profiling	Gives the user access to the content profiling functionality in the Experience Editor and in the Content Editor.
Analytics Maintaining	Gives the user access to the Marketing Control Panel, the Engagement Plan Designer, and the Supervisor. The role gives the user permissions to create goal or page event messages and campaigns for messages.
Analytics Management Reporting	Gives the user access to view the management reports for optimization efforts. This role is typically given to users working with optimization, who wants to view management reports for the optimization efforts. The user can still perform tests, but this is not their main objective.
Analytics Personalization	Gives the user access to the personalization functionality in the Experience Editor and in the Content Editor. Members of this role can create and edit personalization rules. Users who are not members of this role can switch personalization variations.
Analytics Reporting	Gives the user access to the Marketing Control Panel, Experience Analytics, Experience Profile, Path Analyzer, and to the Engagement Plan Monitor.
Analytics Testing	Gives the user access to the Test Lab in the Marketing Control Panel and to the test functionality in the Experience Editor and in the Content Editor. Members of this role can create and edit test variations.
Author	Gives the user access to content in the content tree. This role provides access to basic item editing features, such as the Media Library and the Content Editor, with a reduced set of tabs on the ribbon. This role also has two of the Sitecore Client roles as members, so if you assign just this role to a user, the Sitecore Client Authoring and Sitecore Client Users roles will be automatically assigned to the user.
Designer	Gives the user read and write access to the areas of the content tree that are required when changing layout details for individual items and groups of items via template standard values, as well as items required when configuring the Experience Editor Design Pane. This role also has two of the Sitecore Client roles assigned to it, so if you assign just this role to a user, the Sitecore Client Designing and Sitecore Client Users roles will be automatically assigned to the user. This role provides access to the Experience Editor Design Pane features and the designer options in the Content Editor. Note This role is not a member of the Author and Authoring roles, so it does not allow users to edit items.
Developer	Gives the user access to content manipulation facilities in the Content Editor, plus all the design and authoring roles normally used by client authors and client designers. It also provides access to more functionality on the ribbon of the Content Editor to allow full development features for users assigned to this role. This role also has access to the Development Tools menu in the Sitecore menu, which gives the user access to further development tools, such as the Package Designer.
ECM Advanced Users	

	Gives the user full access to all the functionality in the Email Experience Manager. This role is a member of the List Manager Editors role. Members of this role can: • Delete a message. • Change the default settings. • Open or edit the engagement plan. • Change the recipient lists of a subscription message. • Save a message as a subscription message template. • Change the target device.
ECM Users	Gives the user access to all the basic functionality in the Email Experience Manager, such as create, send, and manage messages. This role is a member of the List Manager Editors role.
Experience Explorer	Gives the user access to the explore mode in the Experience Editor and to manage the *Presets* of the Explore mode in the Marketing Control Panel. The role is intended for marketers who set up campaigns and personalization.
Facebook Message Reviewer	Gives the user access to edit Facebook messages.
List Manager Editors	Gives the user access to the List Manager application for marketers who need to manage contact lists. This is primarily given to EXM users. Members of this role are: • Sitecore\ECM Users • Sitecore\ECM Advanced Users
Sitecore Client Account Managing	Gives the user access to maintain users, roles, and domains in the Access Manager, the Domain Manager, the Role Manager, and the User Manager.
Sitecore Client Advanced Publishing	Gives the user access to the publishing functionality in the Experience Editor and in the Content Editor. This role has access to republish in addition to the same access rights as the Sitecore Client Publishing role.
Sitecore Client Authoring	Gives the user access to basic item editing features. The role is intended for client users to allow access to basic authoring features, such as copy, paste, and rename. The role only influences the Content Editor commands available, it does not influence the Desktop interface's menu or the Control Panel commands.
Sitecore Client Bucket Management	Gives the user access to the Configure tab in the Content Editor and functionality related to Item Buckets.
Sitecore Client Configuring	Gives the user access to the Content Editor features that allow a user to change the configuration details associated with items, such as the icon associated with the item and whether the item is protected or hidden. This role adds the Configure tab to the Content Editor and displays the Appearance, Masters, and Attribute groups.
Sitecore Client Designing	Gives the user access to Experience Editor Design pane features that allow a user to set layout details associated with items in the Sitecore client.
Sitecore Client Developing	Gives the user access to the Developer tab in the Content Editor and functionality commonly required by developers, such as serializing and indexing.
Sitecore Client Forms Author	Gives the user access to the minimum features of the Web Forms for Marketers module. This role allows the user to: • Insert a new form • Edit an existing form • View the Summary report
Sitecore Client Maintaining	Gives the user access to template editing features and reporting tools. This role is intended for administrators and developers.
Sitecore Client Publishing	Gives the user access to the publishing functionality in the Experience Editor and in the Content Editor. Users that are not members of the Publishing role may still be able to publish, but only via automatic publishing features associated with Workflows. This role has access to Incremental publish, Smart publish, and Publish related items.
Sitecore Client Securing	Gives the user access rights to security features in the Content Editor and other relevant applications. This role is intended for users who need to maintain users and access rights.

Sitecore Client Social Authoring	Gives the user access to create messages and post them to social networks. The role is a member of the Sitecore\Author role.
Sitecore Client Translating	Gives the user access to Sitecore’s translation features, such as the command *Scan the database for untranslated fields*. The role is intended for content authors who need access to languages other than the site's default language.
Sitecore Client Users	Gives the user minimal access to Sitecore. With this role, the user can log in to the Sitecore Desktop, but will not have access to any applications. All of the other Sitecore client roles are members of the Sitecore Client Users role, which means that users in any Sitecore client role are automatically members of the Sitecore Client Users role.
Sitecore Limited Content Editor	Limits the amount of Content Editor functionality provided by the Sitecore Client Authoring role (which is still required for users given this role). When a content author is assigned this role, they only have access to the Home, Review, and Publish tabs on the Content Editor ribbon and have no access to copy, move, or sort from the item's right-click menu.
Sitecore Limited Page Editor	Restricts the amount of functionality that is available in the Experience Editor. However, unlike the Sitecore Minimal Page Editor role, users assigned this role see a simple version of the standard Experience Editor ribbon. This role limits the amount of functionality provided by the Sitecore Client Authoring role (which is still required for users given this role), but allows more functional access than the Sitecore Minimal Page Editor role.
Sitecore Local Administrators	Sitecore local administrators can log in to Sitecore and manage the security applications (including assigning security) within that domain. A local administrator cannot create domains or associate domains to users. Note The local administrator role is a member of the Sitecore Client Users, Sitecore Client Account Managing, and Sitecore Client Securing roles. You can use this role as a shortcut to adding these roles to a user.
Sitecore Marketer Form Author	This role inherits access rights from the following roles, and can be used as a shortcut to assign a user all the access rights from those roles: • Sitecore Client Form Author • Analytics Maintaining • Analytics Reporting
Sitecore Minimal Page Editor	Restricts the amount of functionality provided in the Experience Editor to the absolute minimum and users who have been assigned this role do not have access to the Experience Editor ribbon. For example, members of the Minimal Page Editor role cannot switch personalization variations. This role limits the amount of functionality provided by the Sitecore Client Authoring role (which is still required for users given this role).
Social Marketer Message Reviewer	Gives the user access to edit social marketer messages.
Social Message Author	Gives the user access to create, edit, and remove all social marketer messages.
Social Message Workflow Editor	Gives the user access to use the *Draft* state and its commands in the Social Message workflow and the Social Marketer Message workflow.
Social Message Workflow Reviewer	Gives the user access to use the *Awaiting Approval* state and its commands in the Social Message workflow. In addition, the user also has access to the *Awaiting Post Review*, *Post Accepted*, and *Post Removed* states and their commands in the Social Marketer Message workflow.
Twitter Message Reviewer	Gives the user access to edit Twitter messages.

Send feedback about the documentation to docsite@sitecore.net.

The user accounts

Sitecore provides a number of default user accounts that you should not change. If you want to have a user with similar authority as one of the default users, then you should create a new user account rather than edit a default user account. Editing a default user account can affect other areas of the security model.

Predefined user accounts

The default user accounts are:

- sitecore\admin — a predefined administrator user.

- sitecore\anonymous — a user who only has access to the login screen of the website. This is used for users who will go on to access the application framework section of the website.
- extranet\anonymous — a user who is viewing the free to access parts of a website.
- default\anonymous — a user who is assigned to an unauthenticated visitor who is viewing a website that does not have an assigned domain.
- creator\owner — a virtual role that refers to the user specified in an item's item/owner field.

  Important

  To protect your system, you must change the password of the *Admin* user. This should be the first change you make to your security system.

The user template

Users are the named individuals that log in to Sitecore websites.

The user account definition come from the items created with the user template, which has the following fields:

- Administrator — if this field is selected, the user can access everything regardless of their security settings.
- CanBoost — reserved for future use.
- ClientLanguage — the language used in the user interface of the Sitecore client.
- ContentLanguage — the language used in the user interface for Sitecore content.
- DefaultItem — defines the item that is displayed by default in Content Editor.
- Email — the user's email address.
- Fullname — the user's full name.
- Password — the user's password, stored as a "password" field type.
- Portrait — an icon or image that represents the user in the Sitecore menu of the Desktop.
- RegionalIsoCode — the regional ISO code used by the user. This affects how numbers, currency, dates, and times are formatted.
- Roles — the roles assigned to this user.
- Start Url — the URL used for the home page of the user when they login.
- Wallpaper — the Sitecore client's wallpaper.

Send feedback about the documentation to docsite@sitecore.net.

Security considerations

Sitecore recommends that you follow all the security hardening instructions described in our documentation. In addition, the way you implement your Sitecore solution has a significant effect on the security of your website and it may require additional security-related coding and configuration.

Sitecore is not responsible for the security of any other software products that you use with your website. We strongly recommend that you install every available service pack and update for all of the software products that you use.

Note

Sitecore follows all the security hardening recommendations for PaaS solutions by default.

General security recommendations

Although Sitecore can run on several different operating systems, we recommend that you use the newest operating systems, supported by Sitecore, with the most up-to-date security features. Use the Windows update/Automatic update service to keep all your client computers and servers up to date with the most recent security updates and service packs.

You should also create a disaster recovery plan to ensure the rapid resumption of services should a disaster occur. The recovery program should include:

- A plan for acquiring new or temporary equipment.
- A plan for restoring backups.
- Testing the recovery plan.

Change the administrator password

As an extra layer of protection, Sitecore recommends that you create a new administrator account, with a unique name, and delete the out-of-the-box administrator account.

Before you deploy your Sitecore installation, you must change the administrator password to a strong password. Changing the password prevents unauthorized users from using the default password to access the admin account.

Enforce a strong password policy

Sitecore leverages the Microsoft ASP.NET Membership Provider as the out-of-the-box user management system. Sitecore recommends that you change the password policies to one that works for your organization.

In the `web.config` file, in the `<membership>` section, you can set the following properties:

- `minRequiredPasswordLength`
- `minRequiredNonAlphanumericCharacters`
- `maxInvalidPasswordAttempts`
- `passwordAttemptWindow`
- `passwordStrengthRegularExpression`

For more information about these properties, see Microsoft's documentation:

- Membership Properties

Separate your content management and content delivery servers

As part of a *defence in depth* strategy, you should aim to reduce the surface area of your deployment.

Sitecore therefore recommends that you deploy separate content management (internal only) and content delivery (internet facing) servers in a production environment. Furthermore, you should not expose your content management environment to the internet.

If you have to expose your content management environment to the internet, you must:

- Use HTTPS to secure the content management server.
- Consider using IP Filtering to allow only *whitelisted* clients to connect to the Content Management environment

  or

- Consider using the *Dynamic IP Address Restrictions* feature that is available in IIS.

Protect the connectionstrings section in the web.config file

Sitecore stores sensitive information in the `web.config` file in the `<connectionStrings>` section.

You should encrypt the `<connectionStrings>` section to prevent this information from being exposed if the `web.config` file is accessed without authorization.

The Microsoft ASP.NET IIS Registration Tool (`aspnet_regiis.exe`) can be used to encrypt this section with the *–pe* or *–pef* options.

Important

The Microsoft ASP.NET IIS Registration Tool uses the machine key to perform the encryption and therefore you must separately encrypt the `web.config` file on each computer that you install Sitecore on.

For more information about ASP.NET IIS Registration Tool, see Microsoft's documentation:

- ASP.NET IIS Registration Tool

MongoDB

MongoDB can be secured at the network level by default. However, you should follow MongoDB's best practices to harden the security of your Sitecore installation.

For more information about MongoDB security, visit:

- http://www.mongodb.com/blog/post/mongodb-security-best-practices
- https://docs.mongodb.com/manual/security/

Join the Sitecore Security Notification mailing list

If you would like to receive security notifications by email, please sign-up for our security notifications.

Send feedback about the documentation to docsite@sitecore.net.

Change the hash algorithm for password encryption

For user management, Sitecore uses the Microsoft ASP.NET membership provider by default.

When you create a new website, you *must* change the weak default hash algorithm (SHA1) that is used to encrypt user passwords to a stronger algorithm.

To change the hash algorithm:

- Open the `web.config` file and in the `<membership>` node, set the `hashAlgorithmType` setting to the appropriate value. We recommend SHA512.

All the supported hash algorithms are listed at CryptoConfig Class.

Important

The Microsoft ASP.NET membership provider does not provide a facility for upgrading to a different hash algorithm after you have created some user accounts. If you change the hash algorithm, existing users can no longer log into the system and must create a new password.

Send feedback about the documentation to docsite@sitecore.net.

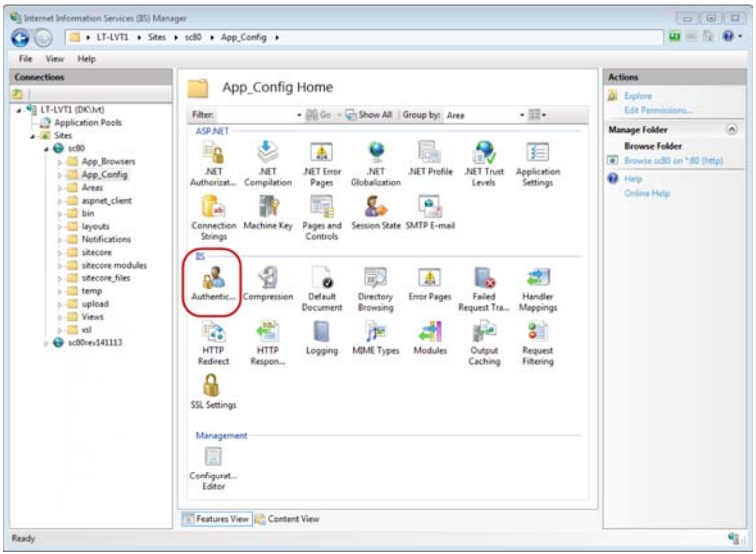
Deny anonymous users access to a folder

You can improve security if you prevent anonymous users from accessing certain key folders. In the Internet Information Services (IIS) manager, you should prevent anonymous users from accessing the following folders:

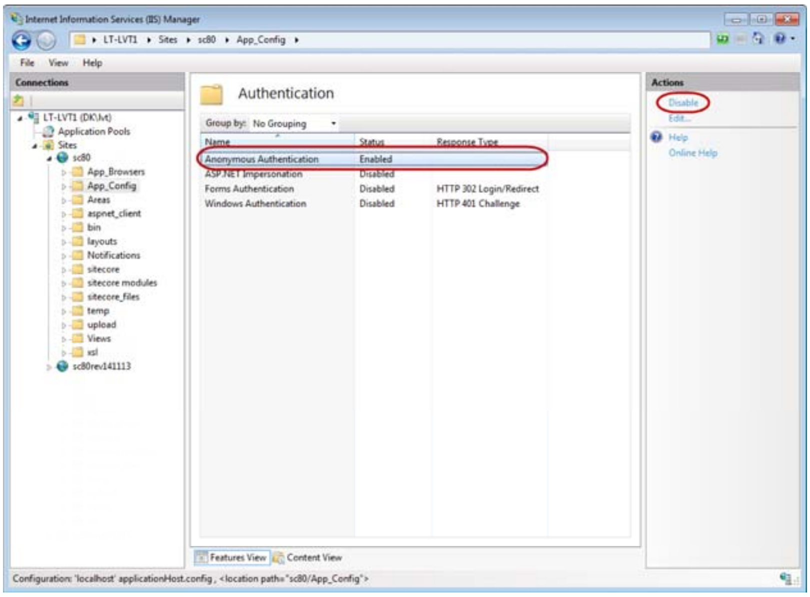
- `/App_Config`
- `/sitecore/admin`
- `/sitecore/debug`
- `/sitecore/login`
- `/sitecore/shell/WebService`

To deny anonymous users access a folder:

1. Open IIS.
2. Navigate to `Web Sites\Default Web Site\App_Config`.
3. In the *App_Config* folder, in the IIS section, double-click Authentication.



4. In the *Authentication* folder, click *Anonymous Authentication* and in the Actions panel, click Disable.



5. Restart IIS.

Repeat this procedure for the *admin* folder (`/sitecore/admin`,) the *debug* folder (`/sitecore/debug`), and the *Webservice* folder (`/sitecore/shell/WebService`).

Send feedback about the documentation to docsite@sitecore.net.

Disable administrative tools

Sitecore contains a number of helpful administrative tools. While these tools can be useful when troubleshooting issues in a production environment, we recommend that you disable them when you are not using them.

Important

You must *never* enable these administrative tools in a Content Delivery environment, or in a Content Management environment that is exposed to the internet.

To disable an administrative tool:

1. In the `<Webroot>/sitecore/admin` folder, locate the relevant file.
2. Add `.disabled` at the end of the existing file name:

   For example:

   Old name: `logs.aspx`

   New name: `logs.aspx.disabled`

You can disable the following ASPX pages:

- Cache.aspx
- DBCleanup.aspx

- dbbrowser.aspx
- ShowServicesConfig.aspx
- eventqueuestats.aspx
- FillDB.aspx
- InstallLanguage.aspx
- Jobs.aspx
- LinqScratchPad.aspx
- Logs.aspx
- MediaHash.aspx
- PackageItem.aspx
- PathAnalyzer.aspx
- Pipelines.aspx
- PublishQueueStats.aspx
- RawSearch.aspx
- RebuildKeyBehaviorCache.aspx
- RebuildReportingDB.aspx
- RedeployMarketingData.aspx
- RemoveBrokenLinks.aspx
- restore.aspx
- SecurityTools.aspx
- serialization.aspx
- SetSACEndpoint.aspx
- ShowConfig.aspx
- SqlShell.aspx
- stats.aspx
- unlock_admin.aspx

The following administrative tools are disabled by default:

- FillDB.aspx
- Unlock_admin.aspx
- SqlShell.aspx

Secure the SqlShell.aspx tool

The `SqlShell.aspx` tool is a very powerful tool for which some extra rules apply.

To control the availability of the `SqlShell.aspx` tool, you can create an empty file in the `<Webroot>/sitecore/admin` folder called `enabled` or `disabled`. This file must not have an extension and does not need to contain any information. These files are not part of the default Sitecore installation.

When you run the `SqlShell.aspx` tool, it checks for these files.

If there is no `enabled` or `disabled` file:

- The tool is available if you are using HTTPS.
- The tool is *not* available if you are using HTTP.

If there is an `enabled` file:

- The tool is available if you are using either HTTPS or HTTP.

If there is a `disabled` file:

- The tool is *not* available if you are using either HTTP or HTTPS.

Important

To prevent anyone from accessing the `SqlShell.aspx` tool, we recommend that you create a `disabled` file.

Send feedback about the documentation to docsite@sitecore.net.

Disable client RSS feeds

If your Sitecore installation contains sensitive information that you want to protect, you can disable the Sitecore client RSS feeds.

RSS technology is designed so that users who follow an RSS link can come directly to the item specified in the URL of the RSS feed. Most RSS readers do not support authentication.

This means that users who subscribe to Sitecore client RSS feeds have direct access to the item specified in the URL of the feed and do not have to identify themselves to the Sitecore security system when they view the feed. However, the Sitecore security system verifies that they are authorized users when they try to perform any actions associated with the client feed.

If an unauthorized user gains access to the URL of a client RSS feed:

- They can follow the link and view all the content contained in the client feed even though their own security permissions do not give them access to this item.
- They cannot perform any actions on the content.
- They cannot view any other content.
- They cannot gain access to the user name or password of the original owner of the client feed.
- They cannot modify the link to gain access to any other content.

Important

Sitecore users should not share client RSS feeds.

To disable Sitecore client RSS feeds:

1. Open the `web.config` file.

2. Locate the `<httpHandlers>` section. Depending on your IIS pool, this section may be called `<Handlers>`.
3. Remove the following handler:

```
<add verb="*" path="sitecore_feed.ashx" type="Sitecore.Shell.Feeds.FeedRequestHandler, Sitecore.Kernel"/>
```

When you remove this handler, you disable all the client feeds that are available inside Sitecore. Any public RSS feeds that you have created are still available to website visitors.

Send feedback about the documentation to docsite@sitecore.net.

Disable SQL Server access from XSLT

Sitecore includes an xslExtension helper for use with SQL Server.

We strongly recommended that you disable the xslExtension helper if:

- You do not need it.
- You are not using Sitecore XSLT renderings.

To disable the xslExtension helper:

1. In the `app_config/include` folder, create a patch file. Give it a file name that ends with the extension `.config`.
2. Insert the following code in the patch file:

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
<sitecore>
<!-- disable XSLT security issue see https://doc.sitecore.net/sitecore_experience_platform/setting_up_and_maintaining/security_ha
<xslExtensions>
<extension type="Sitecore.Xml.Xsl.SqlHelper, Sitecore.Kernel">
<patch:delete/>
</extension>
</xslExtensions>
</sitecore>
</configuration>
```

Send feedback about the documentation to docsite@sitecore.net.

Disable WebDAV

Web Distributed Authoring and Versioning (WebDAV) provides support for remote web authoring. One of the benefits of WebDAV is that it allows you to manage resources from within a browser (Internet Explorer only).

WebDAV extends the HTTP protocol and provides additional functionality and it therefore expands the surface area of your deployments.

Note

You must never enable WebDAV on Sitecore content delivery servers. Only use WebDAV on content management servers (CM).

You should consider whether you need WebDAV functionality in your Content Management environment. If you do not need it, it is best practice to disable it on the CM, to reduce the surface area of your Sitecore environments.

To disable WebDAV:

1. In the `<website>\App_Config\Sitecore\CMS.Core` folder, locate the `Sitecore.WebDAV.config` file.
2. Rename the `Sitecore.WebDAV.config` file to `Sitecore.WebDAV.config.disabled`.

Note

In Sitecore 8.0 - 8.2, the `Sitecore.WebDAV.config` file is stored in the `<website>\App_Config\Include` folder.

Send feedback about the documentation to docsite@sitecore.net.

Improve the security of the website folder

After you install Sitecore, you can increase the security if you place the following folders outside the */Website* root folder:

- */data*
- */indexes*

Important

After moving the */data* folder, you must edit the `web.config` file to point to the new location and configure the permissions for the ASP.NET requests.

Depending on how you install Sitecore, the folders may already be located outside the */Website* folder. If you install Sitecore using:

- The installation program, then the */data* folder is created outside the website root folder and the `web.config` file is edited to point to this location. The */indexes* folder is placed in the */data* folder.

  This is the recommended configuration and you do not need to make any changes.

- A `.zip` file, then the */data* folder is created outside the website root folder, but the `web.config` file is not edited to point to this location. The */indexes* folder is placed in the */data* folder. When you run Sitecore for the first time, it creates another */data* folder in the */Website* folder.

  If you choose to install in this way, you should also edit the `web.config` file to point to the correct location.

  The `web.config` file should look like this:

```
<sitecore database="SqlServer">
  <sc.variable name="dataFolder" value="C:\Inetpub\wwwroot\SitecoreWebsite\data\" />
  <sc.variable name="mediaFolder" value="/upload" />
  <sc.variable name="tempFolder" value="/temp" />
```

Send feedback about the documentation to docsite@sitecore.net.

Increase login security

You can improve the security of the Sitecore login webpage, in the following two ways:

- [Make the login page available only to SSL requests]()
- [Turn off auto complete of user names]()
- [Disable the Remember Last Logged In User Name setting]()

Make the login page available only to SSL requests

To configure the Sitecore Experience Platform to use only SSL requests for the Sitecore login page:

- Create a custom redirect processor that redirects from *http://hostname/sitecore/login* to *https://hostname/sitecore/login*, and redirect all other pages from https to http.

  Use the following code as an example:

```
public class SslLogin
    {
        public void Process(PipelineArgs args)
        {
            string absUrl = HttpContext.Current.Request.Url.AbsoluteUri;
            string localUrl = HttpContext.Current.Request.Url.LocalPath;

            if (localUrl.StartsWith("/sitecore/login") && absUrl.StartsWith("http://") && !Context.IsLoggedIn)
            {
                HttpContext.Current.Response.Redirect(absUrl.Replace("http://", "https://"));
                return;
            }
            if (!localUrl.StartsWith("/sitecore/login") && absUrl.StartsWith("https://") && Context.IsLoggedIn)
            {
                HttpContext.Current.Response.Redirect(absUrl.Replace("https://", "http://"));
            }
        }
    }
```

Turn off auto complete of user names

You can specify that Sitecore should not complete the user name of users automatically when they log in. This is useful, for example, if you do not want user names to be disclosed when content authors log into Sitecore on a shared or public computer. In addition, you can disable the Remember me check box.

To disable auto complete of user names:

- Open the `web.config` file and set the `Login.DisableAutoComplete` setting to true. This disables auto complete on the Sitecore login forms on the `/sitecore/login/default.aspx` and `/sitecore/admin/login.aspx` pages.

To disable the Remember me check box on the login page:

- Open the `web.config` file and set the `Login.DisableRememberMe` setting to *true*. This also ignores any existing *Remember Me* cookies, and all users have to log in again.

Disable the *Remember Last Logged In User Name* setting

As an additional *defense in depth* strategy, Sitecore recommends that you disable the browser's ability to remember the user name of the last logged in user.

To disable the *Remember Last Logged In User Name* setting:

- Open the `sitecore.config` file and set the `Login.RememberLastLoggedInUserName` setting to *false*.

Send feedback about the documentation to docsite@sitecore.net.

Limit access to .XML, .XSLT, and .MRT files

To improve the security of your Sitecore installation, you must edit the `web.config` file. This file is stored in the *Website* folder of your installation, for example at: `C:\Inetpub\wwwroot\YourWebsite\Website`.

To limit access to `.XML`, `.XSLT`, and `.MRT` files:

1. Open the `web.config` file.
2. In the `<system.webServer><handlers>` section, add the following lines:

```
<system.webServer>
  <handlers>
    <add path="*.xml" verb="*" type="System.Web.HttpForbiddenHandler" name="xml (integrated)" preCondition="integratedMode"/>
    <add path="*.xslt" verb="*" type="System.Web.HttpForbiddenHandler" name="xslt (integrated)" preCondition="integratedMode"/>
    <add path="*.config.xml" verb="*" type="System.Web.HttpForbiddenHandler" name="config.xml (integrated)" preCondition="integra
    <add path="*.mrt" verb="*" type="System.Web.HttpForbiddenHandler" name="mrt (integrated)" preCondition="integratedMode"/>
```

This restricts access to all `.XML`, `.XSLT`, and `.MRT` files.

To allow a specific file path to be accessed in an unrestricted manner, such as, `/sitemap.xml`:

1. Open the `web.config` file.
2. In the `<system.webServer><handlers>` section, *before* the handlers that limit access, add the following line:

```
<add path="sitemap.xml" verb="GET" type="System.Web.StaticFileHandler" name="xml allow" />
```

Send feedback about the documentation to docsite@sitecore.net.

PhantomJS and security hardening

Sitecore uses PhantomJS to generate screenshots of pages that you can use in for example email messages.

PhantomJS is an external executable (`.exe`) file that is shipped with Sitecore in the `$(dataFolder)\tools\phantomjs` folder. Sitecore references this file through a setting named `ContentTesting.PhantomJS.ExecutablePath` in the `App_Config\Include\ContentTesting\Sitecore.ContentTesting.config` configuration patch file.

Note

To limit access to the PhantomJS executable file, you should place the `data` folder outside the `webroot` folder.

The Windows process that Sitecore runs under requires the `Read & execute` permission on the executable file. The process also requires `Read` access to the folder itself to access the other files within the folder.

Because PhantomJS is executed from within Sitecore, it runs with the same permissions that have been set for the process that Sitecore is running under. You should run Sitecore with *least* privilege – only allow access to files and folders that Sitecore needs access to.

Sitecore passes PhantomJS a script that is generated from the `RenderScripts` processor in the `getScreenShotForURL` pipeline. This script instructs PhantomJS to write generated images to a specific folder. The `GenerateFilename` processor is responsible for generating the path and file name for use in the script.

By default, the output path is `$(webroot)\temp\screenshots`. The process that Sitecore runs under should already have `Read` and `Write` permissions to the `temp` folder. The output folder location is determined within the pipeline, which you can modify to output the files to a different location.

Note

If the folder that is used for the screenshot files has been changed, you can check the pipeline for the new location.

Increase security on content delivery servers

Because screenshots are used in the UI and emails, it is not necessary to generate them on content delivery servers.

To increase security on the content delivery servers:

- Remove the `$(dataFolder)\tools\phantomjs` folder including the `phantomjs.exe` executable file.
- Disable the `getScreenShotForUrl` pipeline that launches the PhantomJS process.

  To disable the `getScreenshotForUrl` pipeline, create a patch file that disables it in the `Sitecore.ContentTesting.config` configuration file:

  For more information about using patch files, see the topic [Use a patch file to customize the Sitecore configuration](#).

Send feedback about the documentation to docsite@sitecore.net.

Protect media requests

The media request protection feature restricts media URLs that contain dynamic image-scaling parameters, so that only server-generated requests are processed. This ensures that the server only spends resources and disk space on valid image-scaling requests.

You can make your solution more secure and use the Sitecore media request protection feature optimally, if you make a small change to the `Sitecore.Media.RequestProtection.config` file.

To optimize the media request protection feature:

- Open the `/App_Config/Include/Sitecore.Media.RequestProtection.config` file and change the `Media.RequestProtection.SharedSecret` setting to a random string.

Note

In a multiserver setup, you must use the same value for the `Media.RequestProtection.SharedSecret` setting on every server. This ensures that dynamic image scaling works correctly in situations where one server generates the image URL and another handles the request.

Send feedback about the documentation to docsite@sitecore.net.

Remove header information from responses sent by your website

You can improve security and save a small amount of bandwidth if you remove the header information from each response sent by your website.

These headers contain a number of infrastructure details about the framework that is used on your website that you do not need to publicize.

This topic describes how to:

- Remove the X-Aspnet-Version HTTP header
- Remove the X-Powered-By HTTP header
- Remove the X-AspNetMvc-Version HTTP header

Note

If you have installed SXA, it removes all this header information.

Remove the X-Aspnet-Version HTTP header

If you remove the X-Aspnet-Version HTTP header information from each webpage, you save a little bandwidth and ensure that you are not publicizing which version of ASP.NET you are using.

To remove the X-Aspnet-Version HTTP header from each response from ASP.NET, add the following code to the `web.config` file.

```
<system.web>
  <httpRuntime enableVersionHeader="false" />
</system.web>
```

For more information, see the dotnetperls website.

Remove the X-Powered-By HTTP header

If you remove the X-Powered-By HTTP header, you are not publicizing which version of ASP.NET you are using.

To remove the X-Powered-By HTTP header from each response from ASP.NET, add the following code to the `web.config` file:

```
  <system.webServer>
<httpProtocol>
      <customHeaders>
        <remove name="X-Powered-By" />
      </customHeaders>
    </httpProtocol>
  </system.webServer>
```

Remove the X-AspNetMvc-Version HTTP header

If you remove the X-AspNetMvc-Version HTTP header, you are not publicizing which version of ASP.NET MVC you are using.

To remove the X-AspNetMvc-Version HTTP header, add the following code to the `Application_Start` method in the `Global.asax.cs` file:

```
protected void Application_Start(object sender, EventArgs e)
{
  MvcHandler.DisableMvcResponseHeader = true;
  // RegisterRoutes etc... and other stuff
}
```

Note

In Sitecore 9:0 or later, the X-AspNetMvc-Version HTTP header is automatically removed by the `<initialize>` pipeline.

Send feedback about the documentation to docsite@sitecore.net.

Secure the file upload functionality

You can strengthen the security of your Sitecore installation if you deny *script* and *execute* permissions for the *upload* folder.

This topic outlines how to:

- Deny Script and Execute permissions to for the upload folder
- Disable the Upload Watcher
- Prevent a user from uploading certain file types

Deny Script and Execute permissions for the upload folder

If you allow users to modify the content of the *upload* folder, you also give them the permissions to place scripts and executable programs in the folder. Executing these scripts and programs can cause an unexpected behavior on the server.

To avoid this, you can deny permissions to run scripts and executable files in the *upload* folder and thereby prevent an uploaded file from being executed on the server side when a user attempts to download it.

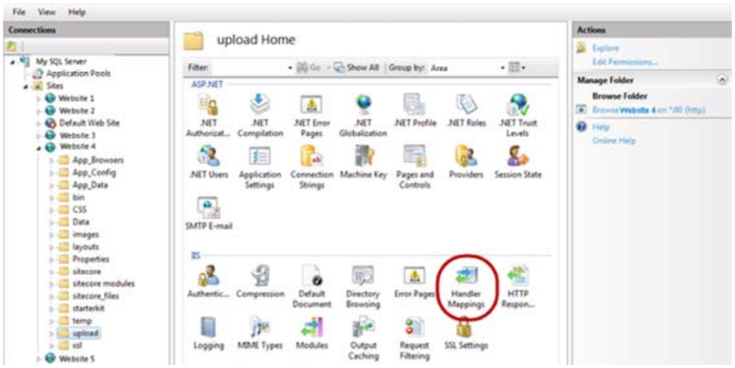
Note

You only need to perform this step if your configuration allows content authors to place files directly in the *upload* folder. For example, if you use a shared directory or FTP server, content authors can quickly place a lot of media in the media library.

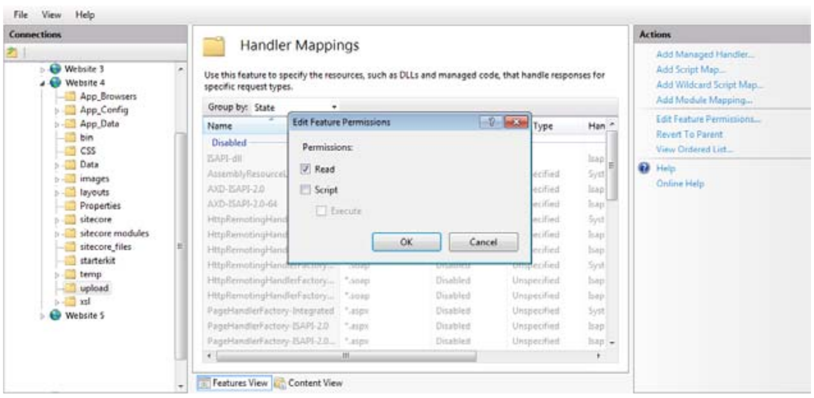
For more information, see the Microsoft website.

To deny both *Script* and *Execute* permissions for the *upload* folder.

1. Open the Internet Information Services (IIS) manager.
2. Navigate to and click the *upload* folder for the relevant website, and then under the IIS section, double-click Handler Mappings



3. In the Actions pane, click Edit Feature Permissions.
4. In the Edit Feature Permissions dialog box, clear the Script and Execute check boxes and click OK.



If your configuration allows content authors to place files directly in the *temp* folder, for example, using a shared directory or an FTP server, you should also deny users *Script* and *Execute* permissions for the *temp* folder.

This also helps to avoid potential security problems, for example, if `.aspx` files are saved in the *temp* folder (for example, from custom code).

Disable the Upload Watcher

To ensure that the only way to upload files to Sitecore is from the Media Library, you should disable the Upload Watcher. This means that you can only upload files from within the Sitecore client and have control over the files that are uploaded.

When you disable the Upload Watcher, files that are placed in the *upload* folder are not automatically uploaded to the Media Library.

To disable the Upload Watcher:

- Open the `web.config` file and remove the following string from the `<system.webServer><modules>` section:

```
<add type="Sitecore.Resources.Media.UploadWatcher,Sitecore.Kernel" name="SitecoreUploadWatcher"/>
```

Prevent a user from uploading certain file types

If you want to have complete control and prevent users from uploading certain file types, for example, `.exe` and `.dll`, you can use the Upload Filter tool.

Download and install the Upload Filter tool

The Upload Filter tool is a Sitecore package that you can download.

The Upload Filter tool contains the following files:

File name	Destination folder
`UploadFilter.config`	`Website\App_Config\Include\`
`UploadFilter.dll`	`WebSite\bin\`

To install the Upload Filter tool:

1. On the Sitecore Launchpad, click Control Panel.
2. Under the Administration section, click Install a Package.
3. Follow the steps in the wizard to first download the package and then install it.

Configure the Upload Filter tool

After you install the Upload Filter tool, you must configure it.

To configure the Upload Filter tool:

1. Open the `UploadFilter.config` file.

```
<processors>
      <uiUpload>
          <processor mode="on" type="Sitecore.Pipelines.Upload.CheckExtension, Sitecore.UploadFilter" patch:before="*[1]">
              <param desc="Allowed extensions (comma separated)"></param>
              <param desc="Blocked extensions (comma separated)">exe,dll</param>
          </processor>
      </uiUpload>
</processors>
```

2. To prevent users from uploading certain file types:
   - In the `Allowed extensions` parameter, enter a comma-separated list of the file extension types that can be uploaded.

     Or

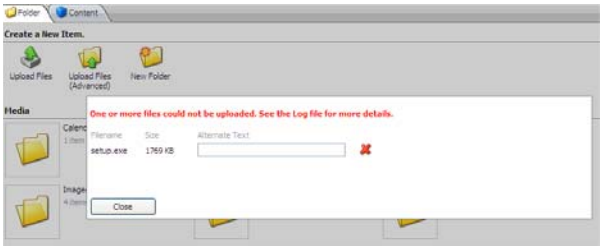
   - In the `Blocked extensions` parameter, enter a comma-separated list of the file extension types that cannot be uploaded.

   You must enter the file extension without the dot.

   Important

   If you set the `Allowed extensions` parameter, the `Blocked extensions` parameter is ignored.

   If you try to upload a file type that is on the blocked list, you see the following message:



Note

After you install the Upload Filter tool, the Upload dialog can freeze if an errors occurs during the upload process. This is a known issue for which there is a solution.

Send feedback about the documentation to docsite@sitecore.net.

Secure the Telerik controls

Sitecore uses some UI controls from Telerik. These controls are only used in a Content Management environment.

Important

If you are running Sitecore 8.2 Update 4 or earlier, you *must first* apply this critical security hotfix.

If you are running Sitecore 8.2 Update 5 or later, you do not need to install this hotfix.

To reduce the attack surface area:

1. In all non-Content Management environments, in the `web.config` file, remove the following nodes:

```
<add name="Telerik_Web_UI_DialogHandler_aspx" verb="*" preCondition="integratedMode" path="Telerik.Web.UI.DialogHandler.aspx" typ
<add name="Telerik_Web_UI_SpellCheckHandler_axd" verb="*" preCondition="integratedMode" path="Telerik.Web.UI.SpellCheckHandler.ax
<add name="Telerik_Web_UI_WebResource_axd" verb="*" preCondition="integratedMode" path="Telerik.Web.UI.WebResource.axd" type="Tel
```

2. In a Content Management environment, you must configure the encryption key that is used to secure the Telerik upload control.

   In the `web.config` file, in the `appSettings` section, create a node for the Telerik configuration encryption keys:

   For example:

```
<appSettings>
<add key="Telerik.AsyncUpload.ConfigurationEncryptionKey" value="YOUR_ENCRYPTION_KEY_HERE" />
<add key="Telerik.Upload.ConfigurationHashKey" value="YOUR_ENCRYPTION_KEY_HERE" />
<add key="Telerik.Web.UI.DialogParametersEncryptionKey" value="YOUR_ENCRYPTION_KEY_HERE" />
</appSettings>
```

   Replace the "YOUR_ENCRYPTION_KEY_HERE" placeholder text with a string of characters that are used to secure the Telerik controls.

   The string should be a set of random characters and numbers, with a maximum length of 256 characters. We recommend that you use a minimum of 32 characters.

For more information, see the Telerik documentation.

Send feedback about the documentation to docsite@sitecore.net.

Use HTTPS on all your Sitecore instances

Using HTTP does not protect data from interception or alteration; therefore, it is best practice to use HTTPS for both your content management and content delivery environments. Sitecore does not include configured HTTPS bindings out of the box.

To enable HTTPS on your Sitecore environments:

1. Ensure you have X.509 certificates from a Certificate Authority.
2. Create the associated bindings on your Sitecore IIS instances.
3. To ensure that all traffic is served over SSL/TLS, open the Sitecore `web.config` file and edit the `<system.web>` section to include these attributes:

```
<system.web>
    <httpCookies httpOnlyCookies="true" requireSSL="true" lockItem="true" />
</system.web>
```

This configuration:

- Ensures that cookies are secure across your site.
- Ensures that a client-side script cannot read the cookies.
- Prevents any additional configuration from overriding these settings.

For more information about configuring HTTPS bindings to an IIS website, visit:

- How to Set Up SSL on IIS 7
- Configuring SSL in IIS Manager

Send feedback about the documentation to docsite@sitecore.net.

Settings supporting UTC implementation

There is one config settings that support date/time types:

- The *Server Time Zone* setting in the `web.config` file.

Server Time Zone

The *Server Time Zone* setting in the `web.config` file specifies the server time zone that is used by the server to convert UTC time to local time, for example "GMT Standard Time".

The value must be a valid Time Zone ID in Microsoft .NET. For more information, see http://msdn.microsoft.com/en-us/library/gg154758.aspx

If the value is blank, Sitecore uses the operating system time zone.

```
<!--  SERVER TIME ZONE
      This setting specifies the server time zone that is  used by the server to convert UTC time to local time, for example "GM
      The value must be parsable to a valid Time Zone ID.              See http://msdn.microsoft.com/en-us/library/gg154758
      If blank, Sitecore uses the operating system time zone.
      Default value: "" (use the operating system time zone)
```

```
-->
<setting name="ServerTimeZone" value=""/>
```

Note

From Sitecore 9.0, the *Analytics.Reports.DisplayDatesInUtc* setting in the `App_Config\Include\Sitecore.Analytics.config` file has been deprecated. The setting specified if dates and times are displayed in UTC or the local server time zone in Experience Analytics reports.

Send feedback about the documentation to docsite@sitecore.net.

Date/time best practices

The following list describes the best practices for configuring date/times in multi-instance CMS solutions spread across different time zones.

- Avoid using local time. Do not use the *DateTime.Now* property, use *DateTime.UtcNow* instead.
- Use only the new methods of `Sitecore.DateUtil` class for date/time conversion operations.
- Convert date/time to universal time using the `ToUniversalTime` method of the `Sitecore.DateUtil` class, before saving date/time to the database.
- Sitecore applications and UI dialogs should display date/time values in the server time zone. You need to convert UTC date/time to server time using the `ToServerTime` method of the `Sitecore.DateUtil` class before displaying date/time values in the UI. This conversion can be done in application code or dialog box logic before the date/time values are assigned to UI controls.
- Use the methods of the `Sitecore.DateUtil` class for parsing strings to date/time and converting date/time to strings.
- When parsing a string to date/time, use `IsoDateToDateTime` or `ParseDateTime` methods of `Sitecore.DateUtil` class. The following overload of the `IsoDateToDateTime` method has the `treatLegacyDatesAsUtc` parameter that allows you to specify whether legacy dates should be treated as UTC dates:

  ```
  public static DateTime IsoDateToDateTime([CanBeNull] string isoDate, DateTime defaultValue, bool treatLegacyDatesAsUtc)
  ```
- When converting date/time to string representation in ISO format, use the `ToIsoDate` method of the `Sitecore.DateUtil` class.
- When comparing two date/time values, make sure that they are both of the same kind. Results of a comparison of two date/time values with different formats will be wrong. Perform a conversion first to make sure that both values are in the same format.
- Keep in mind that the public API (such as created, updated dates of item, etc.) returns universal time, so the client code should handle universal time appropriately.
- When working with the date picker and time picker controls, make sure that the value assigned to the *Value* property of these controls is the server time ISO date. The `IsoDateToServerTimeIsoDate` method of the `Sitecore.DateUtil` class can be used to convert UTC ISO date to server time ISO date.
- When saving input from the date picker and time picker controls to the database, make sure that the date/time value selected in the control is converted to UTC before saving. The `IsoDateToUtcIsoDate` method of `Sitecore.DateUtil` class can be used to convert server time ISO date to UTC ISO date.
- Use the `Sitecore.Utc` and `Sitecore.ServerTime` attributes to set the `DateTime` properties. `DateTime` parameters and methods that return `DateTime` are handled in a specific Sitecore installation. These attributes provide additional information to developers about the expected kind of date/time.

Send feedback about the documentation to docsite@sitecore.net.

Date/time conversions

The Sitecore UTC implementation handles multi-instance CMS solutions that are spread across different time zones. There are both methods and attribute settings that control how date/time values are displayed.

Methods

The following helper methods in the `Sitecore.DateUtil` class handle all new date/time conversion operations:

- `.ToUniversalTime` – converts server date and time to universal time.
- `.ToServerTime` – converts universal date and time to the server time.

Attributes

You can use the following attributes to work with date/time conversions:

- `Sitecore.Utc` – indicates that the selected symbol is a UTC date/time with `DateTime.Kind` set to `DateTimeKind.Utc`.

`Sitecore.ServerTime` – indicates that the selected symbol is a date/time in the server time zone with `DateTime.Kind` set to `DateTime.Kind.Unspecified`.

Send feedback about the documentation to docsite@sitecore.net.

Date/time in cloud installations

In cloud installations, you can configure all the servers to use the same time zone.

In case of failover, where the new server is in a different cluster, the new cloud cluster will display dates in its sever time zone and this may be a different time zone than the one used by the original cluster. By setting all servers to the same time zone, you can avoid confusion in dates that might display differently from one report to another or may seem to be out of sequence.

Send feedback about the documentation to docsite@sitecore.net.

Date/time types

There are several different date/time types that can be displayed in the Sitecore UI or in reports. When you are working in a multi-instance, server environment that crosses multiple time zones, you should use these settings to ensure that your users see date and times consistently.

Local date/time

Local date/time is represented by an instance of the `System.DateTime` class that has a *DateKind* property equal to *DateTimeKind.Local*. This date/time is tied to the local time zone of the computer. It is returned by the *System.DateTime.Now* property. If you have a multi-instance CMS solution, you should avoid using local date/time.

Universal date/time

Universal date/time is represented by an instance of the `System.DateTime` class that has a *DateKind* property equal to *DateTimeKind.Utc*. This date/time is Coordinated Universal Time (UTC). All date/time values should be converted to universal time before saving them to the database. Converting to UTC ensures that users will see consistent date/time values no matter which server they are currently connected to in a multi-instance server environment.

Server date/time

Server date/time is represented by an instance of the `System.DateTime` class that has a *DateKind* property equal to *DateTimeKind.Unspecified*. This is a setting that lets administrators specify the server time zone. All date/time values stored in the database should be UTC, but date/time values in the User Interface (UI) should be displayed in server time. This means that all Sitecore applications and dialogs that display date/time values from the database and/or allow users to edit date/time values should be updated.

If an application displays data from the database, UTC date/time values should be first converted to server time in backend code and then server time values should be displayed in the UI. If the application allows users to enter or edit date/time values, new date/time values should be first converted from server time to UTC and then saved to the database.

Send feedback about the documentation to docsite@sitecore.net.

Storing date/time values in databases

In Sitecore version 8.0 and later, only UTC date/time values should be saved to the database. There can be legacy date/time values in existing Sitecore databases and you need to be able to distinguish UTC date/time values and legacy date/time values. So, for string representations of date/time values, 'Z' is added to the existing format to indicate UTC (for example, '20131031T083552Z'). The methods of the `Sitecore.DateUtil` class responsible for parsing and converting date/time values to strings have been updated and can handle the 'Z' notation correctly.

Sitecore databases store date/time values not only as string values. There are tables that contain `datetime` columns, but they might not store information about time zone. Therefore, all dates in the databases should be treated as UTC and date/time kind should be explicitly set to UTC when reading date/time values from the database.

The following is an example of reading a date/time value from the database:

```
DateTime startDate = reader.InnerReader.GetDateTime(13).SpecifyKind(DateTimeKind.Utc);
```

Send feedback about the documentation to docsite@sitecore.net.

The methods to process date/time in analytics reports

In Engagement Analytics reports, you can display date/time values, either in UTC or in the local server time zone.

The `Analytics.Reports.DisplayDatesInUtc` setting in the `Sitecore.Analytics.config` file controls whether dates display in UTC or local server time. If the value of this setting is true, date/time values are displayed in UTC in reports; otherwise, date/time values are displayed in the local server time zone in reports. The default value of this setting is false.

The `Sitecore.Analytics.Reports.ReportDateInterpreter` class' methods should be used to process date/time values in the Engagement Analytics reports:

Method	Description
`ToUniversalTime(DateTime)`	Converts date and time from UI to UTC, which is acceptable for the database. Use this method to convert report range date/time values provided by the user to filter out the results from the database.
`ToUITime(DateTime)`	Converts UTC date/time from the database to a value acceptable for UI. Use this method to display date/time values from the database in the UI.
`GetDefaultUIStartDate()`	Provides default start dates for the report range. Use this method to initialize report range dates in the UI.
`GetDefaultUIEndDate()`	Provides default end dates for the report range. Use this method to initialize report range dates in UI.
`GetStartDateInclusive()`	Includes start dates in the report range. Use this method to process report range dates provided by the user.

`GetEndDateInclusive()`	Includes end dates in the report range. Use this method to process report range dates provided by the user.

Send feedback about the documentation to docsite@sitecore.net.

Experience Platform terms and concepts

In the Sitecore Experience Platform and Experience Database (xDB), there are several new concepts that need definition.

Experience Platform

The Sitecore Experience Platform is the combination of Sitecore CMS, the xDB, and Experience Marketing applications, such as the Marketing Control Panel.

Experience Database (xDB)

The xDB collects all online and offline customer interactions from all channel sources in a central database repository (the collection database).

The xDB enables you to collect, connect, and analyze all your interaction data to create a comprehensive, unified view of each individual customer. Sitecore applications, such as the Experience Profile and Experience Analytics reports make this data available to marketers, enabling them to manage the customer experience in real time.

Experience Profile

The Experience Profile presents key information about a contact in a single application for sales and marketing employees to gain an insight into the customer experience of your contacts.

The Experience Profile enables you and your sales teams to monitor the key areas of customer experience and interaction, such as visits, campaigns, goals, profiles, outcomes, automations, and keywords. For example, you can see at a glance which events and goals a contact has triggered and how many engagement value points they have accumulated.

In addition, the Experience Profile can easily be extended with additional custom tabs to display business-specific data depending on your requirements.

Experience marketing

Experience marketing describes how you can optimize the customer experience by applying insight about customer behavior in order to increase sales. You do this by collecting data about contacts and interactions, analyzing the data, and acting appropriately according to your findings, for example, by customizing the experience to match the needs of the individual customer.

Interactions can be online or offline. An online interaction can involve a visit to a website, a social media post, or opening an email campaign message. An offline interaction could be a visit to a physical store, a phone call, or a face-to-face meeting.

The way that you manage these interactions is very important. Effective experience marketing enables you to build valuable relationships with your contacts and customers, with the ultimate aim of creating lifelong customers.

Contact

Contacts start out in the system as anonymous visitors to a website, until they become contacts, and then customers. In the xDB, you can use the contact entity to build up a picture of all your visitors, contacts, and potential customers. By collecting data from a wide variety of sources, over time, you can form a detailed picture of your contacts, their behavior, and how they interact with your organization. A contact uses one or more devices to interact with your organization.

In Sitecore xDB, the contact entity is like a container for storing information about the behavior of all your website visitors, customers, and potential customers. This can be data on the devices they use, all their online interactions, social network information, email campaigns, and outcomes. In fact, the xDB has the potential to record all interactions both online and offline, for example, to link to an external CRM or ERP system containing all offline interactions.

Device

A contact uses a device to interact with your organization. For example, a device could be a web browser, smart phone, iPad, or tablet. Details of the device are stored in the collection database in the device entity, which is associated, along with the contact, with a specific interaction.

One or more devices can be associated with a single contact and a single device can be associated with one or more contacts.

Note

Each time a contact accesses the website using a different browser, it creates a new session and appears as a new device in the collection database.

Interaction

An interaction is the way in which a contact or customer communicates or engages with an organization. It is usually a two-way exchange of communication and commitment that takes place via a specific channel. Two-way interactions usually require a higher level of commitment and trust than just viewing a webpage. For example, a contact can provide an email address in return for access to a special offer or campaign from your organization.

Examples of online interactions:

- A visit to a campaign-landing page.
- A visit to your website that originated from a link on a social media site.
- Clicking *Like* on a social media campaign page.
- A visit generated from a link in an email campaign message.
- Opening an email from an email campaign.

From the first anonymous visit to a customer showing high levels of engagement and commitment, all interactions are stored in the collection (MongoDB) database. This record of all online interactions is known as the customer experience.

Channel

When a contact interacts with your website or brand, the channel defines where the contact came from. There are two types of channels – online or offline. Examples of channels could be direct, email newsletter, search non-branded, mobile app, or offline interactions, and channels are divided into groups, such as email, to make them easier to organize. In the Sitecore Experience Platform, when you create campaigns in the *Marketing Control Panel*, you can classify them by setting a channel from the marketing taxonomy on the campaign item. The marketing taxonomy can easily be customized to suit the needs of your organization or business.

Online visit

An online visit is a type of interaction in which a contact visits a website using a device, such as a web browser or mobile phone.

Outcome

An outcome is a business-significant result of a dialogue between a contact and a brand. Through a contact's path to becoming a customer, you can track the events they have triggered and the goals they have converted on their journey. This provides you with an insight into how your contacts interact with your website, as well as the relative financial value they have for your organization.

With outcomes, you can quantify your contacts based on the financial value they bring to your organization rather than by engagement value. Financial value can be based on monetary value, but you can also define financial value more broadly in terms of the value the customer brings to your organization.

Collection database

The collection database is the central data repository for the entire customer experience. It is a highly scalable MongoDB (NoSQL) database that is capable of storing vast amounts of customer experience data. Collection data includes information on contacts, devices, interactions, history, and automations. It can also include data on website visits, responses to email campaigns, and social media interactions.

You can categorize data stored in the collection database in the following ways:

- Customer experience data – for example, data related to contacts and interactions.
- Experience management data – for example, marketers use the *Marketing Control Panel* and other marketing applications to manage campaigns, create goals, mv tests, personalization, and engagement plans. This data is also stored in the collection database.

Note

The collection database does not store the definition items for profiles, goals, campaigns, and other Experience Marketing items that you have created; it only stores a reference to them.

In the xDB, the collection database is a MongoDB database by default.

Reporting database

The reporting database contains aggregated data from the collection database (MongoDB), which can be queried by the Reporting Service. Sitecore reporting applications, such as Experience Analytics reports, use the results of these queries to create dashboards and reports.

In the xDB, the reporting database is a Microsoft SQL Server database by default.

Fact tables

In the xDB, fact tables contain the data (the measurements and metrics) that you want to analyze. Fact tables are similar, but not the same, as cache tables in the SQL reporting database, used to improve the performance of applications, such as Experience Analytics or ECM.

In the xDB, fact tables also contain aggregated data from the collection database that is consumed by the Reporting API and used by Sitecore xDB reporting applications.

Dimension tables

Dimension tables are companion tables to fact tables. They contain descriptive attributes or textual fields that help you understand and analyze the data in fact tables. Dimension tables are similar to lookup tables in the SQL Server reporting database. SQL Server uses a script to update the dimension table with data from fact tables.

Aggregation

The aggregation layer groups and reduces data from the collection database (MongoDB) and then stores it in the reporting SQL Server database. Specifically, this means that a processing or aggregation server converts data into a form that is easier for the Reporting Service to query, for use by applications, such as the Engagement Analytics reports.

By default, the aggregation server is a Sitecore CMS server without a client.

Rebuild of the reporting database

The rebuilding of the reporting database is the re-processing of interactions that have already been aggregated into the reporting database for use by Sitecore reporting applications. You can perform a rebuild after using the conversion tool or after re-classifying a search keyword or traffic type.

Reporting Service

The Reporting Service queries your databases and sends the results back to the Sitecore reporting applications. Report queries can retrieve data from both SQL Server and MongoDB databases. The Reporting Service queries the SQL Server reporting database for grouped data, aggregated data, and trends. It queries the collection database for individual contact or interaction data. The type of query you create depends on the type of data that you need in your reporting applications.

Session state server

The session state server hosts a database that contains all the information required to serve a session, for example, identifying the contact, previous behavior of the contact, and characteristics of the contact. By default, all session state information is stored in the collection database (MongoDB) running on the content delivery cluster but it is also possible to use SQL Server for your session state database.

Private session state

Private session state stores all the data related to a visit or interaction. For example, pages visited, duration of visit, and page events triggered during the visit.

Private session state includes:

- Visit or interaction details
- Pages
- Page events

Note

Private session state is essentially the same ASP.NET session state.

Shared session state

Shared session state is all the data related to a contact and the devices used by the contact to visit a website. This data is stored in a session state database and can be shared across multiple contact sessions on the same server or cluster of servers. For example, each time a contact uses a different browser or device this opens a new session. Shared session state data can be shared across all open sessions.

Shared session state includes:

- Contact
- Device
- Engagement states

Send feedback about the documentation to docsite@sitecore.net.

Configuration patch file examples

The following examples of patch syntax illustrate how patching affects Sitecore configuration. All examples show the original configuration, then the patch files, and finally the configuration after the patch files have been merged with the `Sitecore.config` file.

The examples illustrate:

- Merging configurations
- Inserting an element before a specific element
- Inserting an element after a specific element
- Adding or updating an attribute

To customize your Sitecore configuration, you can use these examples of patch syntax to help you add or change configuration settings in Sitecore.

Merging configurations

Configurations from different patch files are merged in the order the patch files are processed.

```
<!-- Sitecore.config -->

<settings>

  <setting name="name" value="Aaron" />

</settings>

<!-- /App_Config/Include/file1.config -->

<settings>

  <setting name="city" value="New York" />

</settings>

<!-- /App_Config/Include/file2.config -->

<settings>

  <setting name="country" value="USA" />

</settings>
```

The configuration used at runtime is:

```
<settings>

  <setting name="name" value="Aaron" />

  <setting name="city" value="New York" />

  <setting name="country" value="USA" />

</settings>
```

Inserting an element before a specific element

You can insert a configuration element before a specific element. This example adds a processor before the processor with the type `test2`, regardless of the position of `test2` in the configuration.

```
<!-- Sitecore.config -->
<test>
  <processor type="test1" />
  <processor type="test2" />
  <processor type="test3" />
</test>
```

```
<!-- Patch file -->
<test>
  <processor type="testA" patch:before = "processor[@type='test2']"/>
</test>
```

The configuration used at runtime is:

```
<test>
  <processor type="test1" />
  <processor type="testA" />
  <processor type="test2" />
  <processor type="test3" />
</test>
```

Inserting an element after a specific element

You can insert a configuration element after a specific existing element. This example inserts a processor after the processor with the type `test2`, regardless of the position of `test2` in the configuration.

```
<!-- Sitecore.config -->
<test>
  <processor type="test1" />
  <processor type="test2" />
  <processor type="test3" />
</test>
```

```
<!-- Patch file -->
<test>
  <processor type="testA" patch:after = "processor[@type='test2']"/>
</test>
```

The configuration used at runtime is:

```
<test>
  <processor type="test1" />
  <processor type="test2" />
  <processor type="testA" />
  <processor type="test3" />
</test>
```

Adding or updating an attribute

You can add a new attribute or update the value of an existing attribute. If the attribute already exists, its existing value is replaced by the value from the patch file. If the attribute does not already exist in `mysite`, it is added to `mysite`. You can use either the `patch` namespace or the `set` namespace to achieve this effect.

This example adds the `domainPath` attribute to the site `mysite,` and sets it to `/home`, and changes the `virtualFolder` attribute to `/sitecore modules/web`:

```
<!-- Sitecore.config -->
<sites>
  <site name="mysite" virtualFolder="/"></site>
 </sites>
```

This patch file uses the `patch` namespace to change the attributes:

```
<!-- Patch file -->
<sites>
<site name="mysite" virtualFolder="/">
<patch:attribute name="domainPath">/home</patch:attribute>
<patch:attribute name="virtualFolder">/sitecore modules/web</patch:attribute>
</site>
</sites>
```

This patch file uses the `set` namespace to achieve the same result:

```
<!-- Patch file -->

<sites>

<site name="mysite" virtualFolder="/" set:domainPath="/home" set:virtualFolder="/sitecore modules/web"></site>

</sites>
```

The final result is the same for both patch files. The configuration used at runtime is:

```
<sites>

  <site name="mysite" domainPath="/home" virtualFolder="/sitecore modules/web"></site>

</sites>
```

Send feedback about the documentation to docsite@sitecore.net.

Configure a collection database server

The collection database is the central repository in the Experience Database for all customer interaction and contact data, and must be capable of storing vast amounts of customer data. Having a dedicated collection MongoDB server for this purpose means that you can expand your storage capacity more easily as your data requirements grow, while at the same time maintaining the performance of your Sitecore solution.

To configure a dedicated collection database server:

1. Install a blank MongoDB database by following the instructions on the MongoDB website.

   MongoDB installation guide for Windows

2. Also read the xDB hardware guidelines and software recommendations.

   Note

   The collection database runs on MongoDB, so it can run without a Sitecore instance. Therefore, when you create a dedicated collection database, there is no need to install a Sitecore instance on the same server.

3. Configure MongoDB as a Windows service and ensure that it is running correctly.

   For instructions, navigate to the *Configure a Windows Service for MongoDB*¶ section on the MongoDB installation page: MongoDB installation guide for Windows.

4. Configure a MongoDB three-server replica set. A standard MongoDB replica set architecture consists of three MongoDB instances (primary, secondary, and arbiter).

   Note

   For production environments, you should not configure a standalone MongoDB instance. Instead, you should configure a replica set consisting of multiple database servers, to ensure automatic failover and data safety.

5. Once you have configured the collection database on a dedicated server, then you must configure all other servers that have an analytics connection string or need to connect to the MongoDB collection database to point to the server using the correct port number.

Send feedback about the documentation to docsite@sitecore.net.

Configure a content delivery server

A content delivery server makes your web content available to your website visitors. You can configure one or more content delivery servers for improved scalability and better performance. If you expect to have high numbers of visitors or want to configure servers in different geographic locations, you can arrange content delivery servers into clusters.

Before you start configuring server roles ensure that you have read the hardware guidelines for the Sitecore Experience Platform (XP), and the Sitecore Experience Database (xDB).

Note

The information in this topic is valid for Sitecore Experience Platform 8.0 Update 3.

In this type of configuration, you use the content delivery server with the tracking feature.

Important

Tracking should be the only xDB analytics feature you enable on a dedicated content delivery server.

This topic includes the following:

- Files to enable or disable to configure a content delivery server
- Changes to configuration file settings
- Files to enable or disable for Lucene
- Files to enable or disable for Solr
- Solr configuration

xDB considerations

Depending on which type of server you want to configure there are several xDB features available that you can enable or disable such as tracking, processing or reporting. Tracking can be relevant when creating a dedicated content delivery server. It enables the tracking of contacts, personalization, goals, outcomes and more. Tracking is also a feature that you can enable in xDB or in CMS only mode.

Important

Tracking should be the only xDB analytics feature you enable on a dedicated content delivery server.

Session state considerations

In xDB, session state enables you to share your contact sessions across all browsers and devices. This is particularly important if you are configuring clusters of content delivery or processing servers. For each server cluster configure private and shared session state and a session state database to ensure that contacts stay connected to the same server until the session ends.

xAnalytics assembly files

If you have implemented custom code that uses ASP.NET Web API attribute routing, to avoid errors we recommend that you disable the following .dll files:

- `Sitecore.ExperienceAnalytics.dll`
- `Sitecore.ExperienceAnalytics.Client.dll`

Search and indexing considerations

Decide which search and indexing provider you want to use. You can choose Lucene, Solr or a combination of the two depending on the business purpose of your website and the type of server architecture you want to implement.

Depending on whether you use Lucene or Solr for search and indexing there is a different set of configuration steps that you need to follow. Before you choose a search provider, first read more about Using Solr or Lucene with Sitecore, as you have several different options.

If you use Lucene follow the steps below and disable all Solr configuration files and if you use Solr follow the steps below and disable all Lucene configuration files.

To configure a content delivery server:

1. Install a standard Sitecore instance on the server you want to use for content delivery. For installation instructions, read the *Installing the Sitecore Experience Platform* guide on the Sitecore Developer Portal website.
2. Remove or restrict access to the client. You do not need the Sitecore client on a content delivery server.
3. In the `/App_Config/ConnectionStrings.config` file, comment out the following connection strings:

   ```
   <add name="master" connectionString="….." />

   <add name="reporting" connectionString="….."/>

   <add name="tracking.history" connectionString="mongodb:….."/>
   ```

   Content delivery servers do not require all connection strings to be active. For example, there are some security benefits in removing the reporting connection string. Either remove these connection strings completely or add comment tags to hide them.

4. Enable or disable configuration files for the:
   - Content delivery server
   - Lucene provider
   - Solr provider
5. Change the configuration settings for the:
   - Content delivery server
   - Solr provider

Important

When you configure xDB, you must synchronize all the servers in your solution to a single reliable time source, for example by means of the Network Time Protocol (NTP).

The aggregation of engagement automation states depends on the system time and changing this can lead to incorrect aggregation results or loss of data.

Files to enable or disable to configure a content delivery server

To configure a content delivery server, ensure that the following configuration files or .dll files are either enabled or disabled.

Note

Only files that are need to be changed from the default configuration are listed here.

Note

List Manager and Path Analyzer should be completely disabled on a content delivery server.

Configuration file	Folder	Enable	Disable
Marketing platform			
`ScalabilitySettings.config` Disabled by default	`App_Config/Include`	✓	
`Sitecore.Analytics.Automation.TimeoutProcessing.config`	`App_Config/Include`		✓
`Sitecore.Analytics.Processing.Aggregation.Services.config`	`App_Config/Include`		✓

`Sitecore.Analytics.Processing.Services.config`	`App_Config/Include`		✓
`Sitecore.Analytics.Reporting.config`	`App_Config/Include`		✓
`Sitecore.Marketing.Client.config`	`App_Config/Include`		✓
`Sitecore.Processing.config`	`App_Config/Include`		✓
`Sitecore.WebDAV.config`	`App_Config/Include`		✓
`SwitchMasterToWeb.config`	`App_Config/Include`	✓	

Note

Configuration files are applied in alphabetical order. Rename the `SwitchMasterToWeb.config` file so that it is applied after other include files that contain references to the `master` database.

Experience Editor

`Sitecore.ExperienceEditor.Speak.Requests.config`	`App_Config/Include`		✓

Experience Explorer

`Sitecore.ExperienceExplorer.Speak.Requests.config`	`App_Config/Include`		✓

Path Analyzer

`Sitecore.PathAnalyzer.Client.config`	`App_Config/Include`		✓
`Sitecore.PathAnalyzer.config`	`App_Config/Include`		✓
`Sitecore.PathAnalyzer.Processing.config`	`App_Config/Include`		✓
`Sitecore.PathAnalyzer.Services.config`	`App_Config/Include`		✓
`Sitecore.PathAnalyzer.StorageProviders`	`App_Config/Include`		✓
`Sitecore.PathAnalyzer.dll`	`Website/bin`		✓
`Sitecore.PathAnalyzer.Client.dll`	`Website/bin`		✓
`Sitecore.PathAnalyzer.Services.dll`	`Website/bin`		✓
`Sitecore.SequenceAnalyzer.dll`	`Website/bin`		✓

Content Testing

`Sitecore.ContentTesting.Processing.Aggregation.config`	`App_Config/Include/ContentTesting`		✓

Experience Analytics

`Sitecore.ExperienceAnalytics.Aggregation.config`	`App_Config/Include/ExperienceAnalytics`		✓

Sitecore.ExperienceAnalytics.Client.config	App_Config/Include/ExperienceAnalytics	✓
Sitecore.ExperienceAnalytics.Reduce.config	App_Config/Include/ExperienceAnalytics	✓
Sitecore.ExperienceAnalytics.StorageProviders.config	App_Config/Include/ExperienceAnalytics	✓
Sitecore.ExperienceAnalytics.WebAPI.config	App_Config/Include/ExperienceAnalytics	✓
Sitecore.ExperienceAnalytics.dll	Website/bin	✓
Sitecore.ExperienceAnalytics.Client.dll	Website/bin	✓
Experience Profile		
Sitecore.ExperienceProfile.config	App_Config/Include/ExperienceProfile	✓
Sitecore.ExperienceProfile.Client.config	App_Config/Include/ExperienceProfile	✓
Sitecore.ExperienceProfile.Reporting.config	App_Config/Include/ExperienceProfile	✓
Federated Experience Manager		
Sitecore.FXM.Speak.config	App_Config/Include/FXM	✓
List Management		
Sitecore.ListManagement.Client.config	App_Config/Include/ListManagement	✓
Sitecore.ListManagement.config	App_Config/Include/ListManagement	✓
Sitecore.ListManagement.Services.config	App_Config/Include/ListManagement	✓
Social Connected		
Sitecore.Social.ExperienceProfile.config	App_Config/Include/Social	✓
Speak		
.Sitecore.Speak.Important.config	App_Config/Include	✓
Sitecore.Speak.AntiCsrf.SheerUI.config	App_Config/Include	✓
Sitecore.Speak.Applications.config	App_Config/Include	✓
Sitecore.Speak.config	App_Config/Include	✓
Sitecore.Speak.ItemWebApi.config	App_Config/Include	✓
Sitecore.Speak.LaunchPad.config	App_Config/Include	✓

`Sitecore.Speak.Mvc.config`	`App_Config/Include`	✓

Note

To create an additional server role to trigger and process automation states by timeout for contacts with live sessions on your website, install the Sitecore Engagement Automation [Live Session Agent](#) module. You can download the module from the [Sitecore Developer Portal](#).

Changes to configuration file settings

In the following configuration files, find the appropriate section and update each of the parameter values.

You can remove references to the master database using the `SwitchMasterToWeb.config` configuration file patch (included in the list of config files to enable).

Note

In Sitecore, you can create your own names for databases and indexes. This table uses the default Sitecore names for all databases and indexes, such as *master*, *web* and *sitecore_marketing_asset_index_web*.

Configuration file	Section
`ScalabilitySettings.config`	`configuration/sitecore/settings/setting name="InstanceName"/patch:at`
`Sitecore.Analytics.config`	`configuration/sitecore/scheduling/<agent type="Sitecore.Analytics.Ta` Comment out this section.

`Sitecore.Analytics.Tracking.config`	`<setting name="Analytics.DefaultDefinitionDatabase" value="master" /`
	
	`<setting name="Analytics.ClusterName" value="default-cd-cluster" />`
	If you are creating a cluster of content delivery servers, use your domain name wher
	For example, `value="cluster1.domain.com"`
	Do this for each node in the cluster. For a single content delivery server, use the don
	
	`<setting name="Analytics.HostName" value="" />`
	The global domain name of the Sitecore website.
	The cluster and domain name should be identical for all content delivery servers tha
	For example:
	`<setting name="Analytics.ClusterName" value="cluster1.domain.com" /:`
	`<setting name="Analytics.HostName" value="www.domain.com" />`

`Sitecore.Buckets.config`	`configuration/sitecore databases/database` Comment out the entire master database section.
`Sitecore.ContentSearch.config`	`configuration/sitecore/scheduling/agent`
`Sitecore.ContentSearch.DefaultConfigurations.config` Comment out references to the master database.	`configuration/sitecore/contentSearch/indexConfigurations/indexUpdate`

Comment out references to the master database.	`configuration/sitecore/contentSearch/indexConfigurations/indexUpdate`
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.config`	`configuration/sitecore/marketingDefinitions/asset/repositories/item` Change the database name from *master* to *web*. `configuration/sitecore/marketingDefinitions/asset/repositories/item`

`Sitecore.Marketing.config`	`configuration/sitecore/marketingDefinitions/campaign/repositories/i`
	In the marketing definition for *Campaigns*, change the database name from *master* t
	`configuration/sitecore /marketingDefinitions/goal/repositories/item`
	In the marketing definition for *Goals*, change the database name from *master* to *web*
	`marketingDefinitions/outcome/repositories/item`
	In the marketing definition for *Outcomes*, change the database name from *master* to

`Sitecore.Social.config`	`configuration/sitecore/settings/setting`

Files to enable or disable for Lucene

To configure a content delivery server when using Lucene for search and indexing, you need to enable or disable the following additional configuration files.

Note

All relevant Lucene and Solr configuration files are included in this section regardless of whether they are enabled or disabled by default.

To disable a configuration file change the extension from *.config* to *.disabled*.

Configuration file	Folder	Enable	D
Marketing platform			
`Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config`	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Analytics.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Core.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Master.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Web.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.DefaultIndexConfiguration.config` Disabled by default	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Analytics.config` Disabled by default	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Core.config` Disabled by default	`App_Config/Include`		✓

`Sitecore.ContentSearch.Solr.Index.Master.config` Disabled by default	`App_Config/Include`	
`Sitecore.ContentSearch.Solr.Index.Web.config` Disabled by default	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Master.config`	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Web.config` Enabled by default	`App_Config/Include`	✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.IndexConfiguration.config` Enabled by default	`App_Config/Include`	✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Master.config`	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Web.config`	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.IndexConfiguration.config`	`App_Config/Include`	
Content Testing		
`Sitecore.ContentTesting.Lucene.IndexConfiguration.config` Enabled by default	`App_Config/Include/ContentTesting`	
`Sitecore.ContentTesting.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/ContentTesting`	
Federated Experience Manager		
`Sitecore.FXM.Lucene.Index.DomainsSearch.config` Enabled by default	`App_Config/Include/FXM`	✓
`Sitecore.FXM.Solr.Index.DomainsSearch.config` Disabled by default	`App_Config/Include/FXM`	
List Management		
`Sitecore.ListManagement.Lucene.Index.List.config`	`App_Config/Include/ListManagement`	
`Sitecore.ListManagement.Lucene.IndexConfiguration.config`	`App_Config/Include/ListManagement`	
`Sitecore.ListManagement.Solr.Index.List.config` Disabled by default	`App_Config/Include/ListManagement`	
`Sitecore.ListManagement.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/ListManagement`	
Social Connected		
`Sitecore.Social.Lucene.Index.Master.config`	`App_Config/Include/Social`	
`Sitecore.Social.Lucene.Index.Web.config` Enabled by default	`App_Config/Include/Social`	✓

`Sitecore.Social.Lucene.IndexConfiguration.config` Enabled by default	`App_Config/Include/Social`	✓	
`Sitecore.Social.Solr.Index.Master.config` Disabled by default	`App_Config/Include/Social`		✓
Note Configuration files using the Master database are not required on a content delivery server.			
`Sitecore.Social.Solr.Index.Web.config` Disabled by default	`App_Config/Include/Social`		✓
`Sitecore.Social.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/Social`		✓

Files to enable or disable for Solr

To configure a content delivery server when using Solr for search and indexing, you need to enable or disable the following additional configuration files.

Note

All relevant Lucene and Solr configuration files are included in this section regardless of whether they are enabled or disabled by default.

To disable a configuration file change the extension from *.config* to *.disabled*.

Configuration file	Folder	Enable	D
Marketing platform			
`Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Analytics.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Core.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Master.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Web.config` Enabled by default	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.DefaultIndexConfiguration.config`	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Analytics.config` Disabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Core.config` Disabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Master.config` Disabled by default	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Web.config` Disabled by default	`App_Config/Include`	✓	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Master.config`	`App_Config/Include`		✓

Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Web.config	App_Config/Include		✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.IndexConfiguration.config	App_Config/Include		✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Master.config	App_Config/Include		✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Web.config	App_Config/Include	✓	
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.IndexConfiguration.config	App_Config/Include	✓	

Note

Configuration files using the *Master* database are not required on a content delivery server.

Content Testing

Sitecore.ContentTesting.Lucene.IndexConfiguration.config	App_Config/Include/ContentTesting		✓
Sitecore.ContentTesting.Solr.IndexConfiguration.config *Disabled by default	App_Config/Include/ContentTesting		✓

Federated Experience Manager

Sitecore.FXM.Lucene.Index.DomainsSearch.config	App_Config/Include/FXM		✓
Sitecore.FXM.Solr.Index.DomainsSearch.config	App_Config/Include/FXM		✓

List Management

Sitecore.ListManagement.Lucene.Index.List.config	App_Config/Include/ListManagement		✓
Sitecore.ListManagement.Lucene.IndexConfiguration.config	App_Config/Include/ListManagement		✓
Sitecore.ListManagement.Solr.Index.List.config Disabled by default	App_Config/Include/ListManagement		✓
Sitecore.ListManagement.Solr.IndexConfiguration.config Disabled by default	App_Config/Include/ListManagement		✓

Social Connected

Sitecore.Social.Lucene.Index.Master.config	App_Config/Include/Social		✓
Sitecore.Social.Lucene.Index.Web.config	App_Config/Include/Social		✓
Sitecore.Social.Lucene.IndexConfiguration.config	App_Config/Include/Social		✓
Sitecore.Social.Solr.Index.Master.config	App_Config/Include/Social		✓
Sitecore.Social.Solr.Index.Web.config	App_Config/Include/Social	✓	
Sitecore.Social.Solr.IndexConfiguration.config	App_Config/Include/Social	✓	

Solr configuration

If you use Solr as your search and indexing provider, first ensure that you have enabled all Solr configuration files required on a content delivery server. If you use multiple cores, no further configuration is necessary.

It is no longer necessary to store each default index in a separate Solr core. However, for optimal performance this approach is preferred. If you store each default index in a separate Solr core the names of these cores will match the names of the Sitecore indexes.

Solr cores required on a content delivery server:

- `sitecore_core_index`
- `sitecore_web_index`
- `sitecore_fxm_web_index`
- `sitecore_marketing_asset_index_web`
- `social_messages_web`

Note

Multiple Solr cores may improve the performance of your search and indexing. For example, you should use separate cores for each index in a production environment and only share indexes on a single core in a development environment. The Solr website provides more detailed guidance and advice when configuring Solr.

Renaming Solr cores

You can change the default Solr core names to your own custom names and can also store several Sitecore indexes in a shared Solr core.

If you want to change the default Solr core names and use your own custom names, then you also need to change the names in each Sitecore Solr configuration file.

To set a Solr core for a Sitecore index:

1. In the `App_Config\Include` folder, open the configuration file where the Sitecore index is declared. For example, in the `Sitecore.ContentSearch.Solr.Index.Core.config` file.
2. Navigate to the section where the index is declared: `configuration/sitecore/contentSearch/configuration/indexes/`:

```
<index id="..." type="Sitecore.ContentSearch.SolrProvider.SolrSearchIndex,
 Sitecore.ContentSearch.SolrProvider">
   <param desc="name">$(id)</param>
   <param desc="core">$(id)</param>
```

3. Change value of the `core` parameter to the custom name of the Solr core you want to use.

   Follow the same procedure for each Solr configuration file where you want to change the Solr core name.

Send feedback about the documentation to docsite@sitecore.net.

Configure a content management server

Content management servers enable content editors to create and publish content to a web site. You have several options when you configure a content management server. It is possible to mix server features, such as, content delivery and content management together on the same server or you could configure a single content management server. You could also have multiple content management servers, depending on your content editing needs.

Before you start configuring server roles ensure that you have read the hardware guidelines for the Sitecore Experience Platform (XP), and the Sitecore Experience Database (xDB).

Note

The information in this topic is valid for Sitecore Experience Platform 8.0 Update 3.

In this type of configuration you use the content management server with the reporting server feature.

Note

If you are configuring a dedicated content management server, you must disable the `Sitecore.Analytics.Tracking.Database.config` file. This means that the content management server is used for tracking but does not save any changes to the collection database. If you want to mix content management with other server features, or if there is a chance that other modules may depend on the tracking data access API, then you must enable the `Sitecore.Analytics.Tracking.Database.config` file.

This topic outlines:

- Files to enable or disable to configure a content management server
- Changes to configuration file settings
- Files to enable or disable for Lucene
- Files to enable or disable for Solr
- Solr configuration
- Configuring the Path Analyzer on a content management server with a remote client

Search and indexing considerations

Decide which search and indexing provider you want to use. You can choose Lucene, Solr or a combination of the two depending on the business purpose of your website and the type of server architecture you want to implement.

Depending on whether you use Lucene or Solr for search and indexing there is a different set of configuration steps that you need to follow. Before you choose a search provider, first read more about Using Solr or Lucene with Sitecore, as you have several different options.

If you use Lucene follow the steps below and disable all Solr configuration files and if you use Solr follow the steps below and disable all Lucene configuration files.

To configure a content management server:

1. Install a standard Sitecore instance on the server you want to use for content management. For installation instructions, read the *Installing the Sitecore Experience Platform* guide on the [Sitecore Developer Portal] website.
2. In the `/App_Config/ConnectionStrings.config` file, configure the following database connection strings:

   SQL Server:

   - core
   - master
   - web
   - reporting

   MongoDB:

   - analytics
   - tracking.live
   - tracking.history
   - tracking.contact

   Note

   Before running history processing on a content management server, using the RebuildReportingDb.aspx page, make sure you add a *reporting.secondary* connection string to the `ConnectionStrings.config` file.

   For example:

   ```
   <add name="reporting.secondary" connectionString="…."/>
   ```

   Note

   You should always configure the *sessions* connection string unless you use the InProc mode for session state. It is ok to configure a content management server with the InProc session provider because it does not perform the same tasks as a content delivery server. This also means a content management server should never be a part of a content delivery cluster.

3. Enable or disable configuration files for the:
   - Content management server
   - Lucene provider
   - Solr provider
4. Change the configuration settings for the:
   - Content management server
   - Solr provider

Important

When you configure xDB, you must synchronize all the servers in your solution to a single reliable time source, for example by means of the Network Time Protocol (NTP).

The aggregation of engagement automation states depends on the system time and changing this can lead to incorrect aggregation results or loss of data.

Files to enable or disable to configure a content management server

To configure a content management server, ensure that the following configuration files or .dll files are either enabled or disabled.

To disable a configuration file change the extension from *.config* to *.disabled*.

Configuration file	Folder	Enable	Disable
Marketing platform			
`ScalabilitySettings.config` Disabled by default	`App_Config/Include`	✓	
`Sitecore.Analytics.Automation.TimeoutProcessing.config`	`App_Config/Include`	✓	
`Sitecore.Analytics.Processing.Aggregation.Services.config`	`App_Config/Include`		✓
`Sitecore.Analytics.Processing.Services.config`	`App_Config/Include`		✓
`Sitecore.Analytics.Tracking.Aggregation.config`	`App_Config/Include`		✓
`Sitecore.Analytics.Tracking.Database.config`	`App_Config/Include`		✓
`Sitecore.Analytics.ExcludeRobots.config`	`App_Config/Include`		✓

`Sitecore.Analytics.Tracking.RobotDetection.config`	`App_Config/Include`		✓

Note

All `Remote.Client.config` files are disabled by default. You should enable these files if you have a reporting server in another location. Disable these files on a content delivery server and enable them on a content management server. The `<httpTransportFactory />` node in the `Sitecore.Xdb.Remote.Client.config` file must contain the base URL of the remote data server. If you are using the Sitecore xDB Cloud 1.0 do not enable the `Sitecore.Xdb.Remote.Client.config` file.

Path Analyzer			
`Sitecore.PathAnalyzer.Processing.config`	`App_Config/Include`		✓
`Sitecore.PathAnalyzer.RemoteClient.config`	`App_Config/Include`	✓	
`Sitecore.PathAnalyzer.StorageProviders.config`	`App_Config/Include`		✓
`Sitecore.Xdb.Remote.Client.config`	`App_Config/Include`	✓	
Content Testing			
`Sitecore.ContentTesting.Processing.Aggregation.config`	`App_Config/Include/ContentTesting`		✓
Experience Analytics			
`Sitecore.ExperienceAnalytics.Aggregation.config`	`App_Config/Include/ExperienceAnalytics`		✓
`Sitecore.ExperienceAnalytics.Reduce.config`	`App_Config/Include/ExperienceAnalytics`		✓
Social Connected			
`Sitecore.Social.ScalabilitySettings.config` Disabled by default	`App_Config/Include/Social`	✓	

Note

The configuration files required for the Content Testing, Experience Analytics, Experience Profile, Federated Experience Manager, List Management and Social Connected applications are enabled by default. Always enable the `Sitecore.Social.ScalabilitySettings.config` file on content management servers in a scaled environment.

Disabling the Sitecore.Analytics.Tracking.RobotDetection.config file

When you configure a content management server the `Sitecore.Analytics.Tracking.RobotDetection.config` is one of the files you disable. However, to completely disable robot detection you also need to make the following changes to the `Web.config` file:

1. In the `system.web/httpModules` node, `name="MediaRequestSessionModule"` change the following line from:

   `"Sitecore.Analytics.RobotDetection.Media.MediaRequestSessionModule, Sitecore.Analytics.RobotDetection"`

   to `"Sitecore.Analytics.Media.MediaRequestSessionModule, Sitecore.Analytics"`.

2. In the `system.webServer/modules node`, `name="MediaRequestSessionModule"` change the following line from: `"Sitecore.Analytics.RobotDetection.Media.MediaRequestSessionModule, Sitecore.Analytics.RobotDetection"`

   to `"Sitecore.Analytics.Media.MediaRequestSessionModule, Sitecore.Analytics"`.

Changes to configuration file settings

In the following configuration files, find the appropriate section and update each of the parameter values.

Configuration file	Section	Parameter	Value
`Sitecore.Analytics.Tracking.config`	`<setting name="Analytics.ClusterName" value="default-cd-cluster" />` If you are creating a content management server cluster assign a name to each content management server that you add.	`Analytics.ClusterName`	`default-cd-cluster`

			Change the default value from cd to cm.
	`configuration/sitecore/settings/` Disable automatic robot detection.	`Analytics.AutoDetectBots`	`false`
`Sitecore.Social.config`	`configuration/sitecore/settings/setting name="Social.LinkDomain"`	`setting name`	`sc8CM`

`ScalabilitySettings.config`	`configuration/sitecore/settings/setting name="InstanceName"`	`patch:attribute name`	`sc8CM1`
			If you have multiple CM servers, give each a unique name, such as:
			`- sc8CM2`
			`- sc8CM3`

`Sitecore.Xdb.Remote.Client.config`	`configuration/sitecore/httpTransportFactory/param desc="serviceUrl"`	`param desc="serviceUrl"`	`http://sc8Rep/`

Files to enable or disable for Lucene

To configure a content management server when using Lucene for search and indexing, you need to enable or disable the following additional configuration files.

Note

All relevant Lucene and Solr configuration files are included in this section regardless of whether they are enabled or disabled by default.

To disable a configuration file change the extension from *.config* to *.disabled*.

Configuration file	Folder	Enable	D
Marketing platform			
`Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config`	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Analytics.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Core.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Master.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Web.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.DefaultIndexConfiguration.config` Disabled by default	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Analytics.config` Disabled by default	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Core.config` Disabled by default	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Master.config` Disabled by default	`App_Config/Include`		✓

Sitecore.ContentSearch.Solr.Index.Web.config Disabled by default	App_Config/Include		✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Master.config Enabled by default	App_Config/Include	✓	
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Web.config	App_Config/Include		✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.IndexConfiguration.config Enabled by default	App_Config/Include	✓	
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Master.config Disabled by default	App_Config/Include		✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Web.config Disabled by default	App_Config/Include		✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.IndexConfiguration.config Disabled by default	App_Config/Include		✓
Content Testing			
Sitecore.ContentTesting.Lucene.IndexConfiguration.config Enabled by default	App_Config/Include/ContentTesting	✓	
Sitecore.ContentTesting.Solr.IndexConfiguration.config Disabled by default	App_Config/Include/ContentTesting		✓
Federated Experience Manager			
Sitecore.FXM.Lucene.Index.DomainsSearch.config Enabled by default	App_Config/Include/FXM	✓	
Sitecore.FXM.Solr.Index.DomainsSearch.config	App_Config/Include/FXM		✓
List Management			
Sitecore.ListManagement.Lucene.Index.List.config Enabled by default	App_Config/Include/ListManagement	✓	
Sitecore.ListManagement.Lucene.IndexConfiguration.config Enabled by default	App_Config/Include/ListManagement	✓	
Sitecore.ListManagement.Solr.Index.List.config	App_Config/Include/ListManagement		✓
Sitecore.ListManagement.Solr.IndexConfiguration.config	App_Config/Include/ListManagement		✓
Social Connected			
Sitecore.Social.Lucene.Index.Master.config Enabled by default	App_Config/Include/Social	✓	
Sitecore.Social.Lucene.Index.Web.config Enabled by default	App_Config/Include/Social	✓	

`Sitecore.Social.Lucene.IndexConfiguration.config` Enabled by default	`App_Config/Include/Social`	✓	
`Sitecore.Social.Solr.Index.Master.config` Disabled by default	`App_Config/Include/Social`		✓
`Sitecore.Social.Solr.Index.Web.config` Disabled by default	`App_Config/Include/Social`		✓
`Sitecore.Social.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/Social`		✓

Files to enable or disable for Solr

To configure a content management server when using Solr for search and indexing, you need to enable or disable the following additional configuration files.

Note

All relevant Lucene and Solr configuration files are included in this section regardless of whether they are enabled or disabled by default.

To disable a configuration file change the extension from *.config* to *.disabled*.

Configuration file	**Folder**	**Enable**	**D**
Marketing platform			
`Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Analytics.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Core.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Master.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Web.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.DefaultIndexConfiguration.config`	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Analytics.config`	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Core.config`	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Master.config`	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Web.config`	`App_Config/Include`	✓	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Master.config`	`App_Config/Include`		✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Web.config`	`App_Config/Include`		✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.IndexConfiguration.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Master.config`	`App_Config/Include`	✓	

Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Web.config	App_Config/Include		✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.IndexConfiguration.config	App_Config/Include	✓	

Note

All `Remote.Client.config` files are disabled by default. You should enable these files if you have a reporting server in another location. Disable these files on a conte delivery server and enable them on a content management server. The `<httpTransportFactory />` node in the `Sitecore.Xdb.Remote.Client.config` file must cor the base URL of the remote data server. If you are using the Sitecore xDB Cloud 1.0 do not enable the `Sitecore.Xdb.Remote.Client.config` file.

Content Testing			
Sitecore.ContentTesting.Lucene.IndexConfiguration.config			✓
Sitecore.ContentTesting.Solr.IndexConfiguration.config	App_Config/Include/Content Testing	✓	
Federated Experience Manager			
Sitecore.FXM.Lucene.Index.DomainsSearch.config	App_Config/Include/FXM		✓
Sitecore.FXM.Solr.Index.DomainsSearch.config	App_Config/Include/FXM	✓	
List Management			
Sitecore.ListManagement.Lucene.Index.List.config	App_Config/Include/ListManagement		✓
Sitecore.ListManagement.Lucene.IndexConfiguration.config	App_Config/Include/ListManagement		✓
Sitecore.ListManagement.Solr.Index.List.config	App_Config/Include/ListManagement	✓	
Sitecore.ListManagement.Solr.IndexConfiguration.config	App_Config/Include/ListManagement	✓	
Social Connected			
Sitecore.Social.Lucene.Index.Master.config	App_Config/Include/Social		✓
Sitecore.Social.Lucene.Index.Web.config	App_Config/Include/Social		✓
Sitecore.Social.Lucene.IndexConfiguration.config	App_Config/Include/Social		✓
Sitecore.Social.Solr.Index.Master.config	App_Config/Include/Social	✓	
Sitecore.Social.Solr.Index.Web.config	App_Config/Include/Social	✓	
Sitecore.Social.Solr.IndexConfiguration.config	App_Config/Include/Social	✓	

Note

If you are using Solr for search and indexing ensure that all Lucene configuration files are disabled.

Solr configuration

If you use Solr as your search and indexing provider, first ensure that you have enabled all Solr configuration files required on a content management server. If you use multiple cores, no further configuration is necessary.

It is no longer necessary to store each default index in a separate Solr core. However, for optimal performance this approach is preferred. If you store each default index in a separate Solr core the names of these cores will match the names of the Sitecore indexes.

Solr cores required on a content management server:

- `sitecore_core_index`
- `sitecore_master_index`
- `sitecore_fxm_master_index`
- `sitecore_list_index`
- `sitecore_marketing_asset_index_master`
- `social_messages_master`

Note

Multiple Solr cores may improve the performance of your search and indexing. For example, you should use separate cores for each index in a production environment and only share indexes on a single core in a development environment. The Solr website provides more detailed guidance and advice when configuring Solr.

Renaming Solr cores

You can change the default Solr core names to your own custom names and can also store several Sitecore indexes in a shared Solr core.

If you want to change the default Solr core names and use your own custom names, then you also need to change the names in each Sitecore Solr configuration file.

To set a Solr core for a Sitecore index:

1. In the `App_Config\Include` folder, open the configuration file where the Sitecore index is declared. For example, in the `Sitecore.ContentSearch.Solr.Index.Core.config` file.
2. Navigate to the section where the index is declared: `configuration/sitecore/contentSearch/configuration/indexes/`:

```
<index id="..." type="Sitecore.ContentSearch.SolrProvider.SolrSearchIndex,
 Sitecore.ContentSearch.SolrProvider">
   <param desc="name">$(id)</param>
   <param desc="core">$(id)</param>
```

3. Change value of the `core` parameter to the custom name of the Solr core you want to use.

   Follow the same procedure for each Solr configuration file where you want to change the Solr core name.

Configuring the Path Analyzer on a content management server with a remote client

If you connect to a dedicated remote reporting instance, then the content management server does not have direct access to the `Sitecore.Analytics` SQL database. Instead, the Path Analyzer client retrieves data from a remote server with direct access to the SQL database using Web API services.

Note

This does not apply to Sitecore xDB Cloud Edition.

To enable the Path Analyzer client to communicate with the remote server, make the following configuration file changes:

Configuration file	Folder	Enable	Disable
`Sitecore.PathAnalyzer.RemoteClient.config`	`App_Config/Include`	✓	
`Sitecore.PathAnalyzer.Processing.config`	`App_Config/Include`		✓
`Sitecore.PathAnalyzer.Services.RemoteServer.config`	`App_Config/Include`		✓

Model view controller (MVC)

When developing Sitecore applications, you can use MVC alongside or as an alternative to Sitecore web forms.

To enable Sitecore MVC with the xDB, on a content management server, make the following configuration file changes:

Configuration file	Folder	Enable	Disable
`Sitecore.MvcAnalytics.config`	`App_Config/Include`	✓	

Send feedback about the documentation to docsite@sitecore.net.

Configure a processing server

You can configure a processing server in much the same way as you configure a content delivery or content management server. You can configure one or multiple processing servers for aggregation or other kinds of processing. Configuring multiple processing servers provides better scalability and performance. As a general rule, the more aggregation server instances you create, the faster aggregation processing will be.

Before you start configuring server roles ensure that you have read the hardware guidelines for the Sitecore Experience Platform (XP), and the Sitecore Experience Database (xDB).

Note

The information in this topic is valid for Sitecore Experience Platform 8.0 Update 3.

In this type of configuration, you use the processing server with one or more of the following processing features.

- *Processing*
- *Aggregation*

This topic outlines:

- Files to enable or disable to configure a processing server
- Changes to configuration file settings
- Files to enable or disable for Lucene
- Files to enable or disable for Solr
- Solr configuration
- Configure aggregation agents or threads
- Configuring the Path Analyzer on a remote processing/reporting shared server

Search and indexing considerations

When you configure a processing server in a scaled environment, it is recommended that you always use Solr because Solr supports calls over HTTP(S). Lucene is a file based indexing system which means if the index is not located on the server that the request is coming from, then you have to ensure that indexes are synchronized to all servers. There are several alternative ways to configure Lucene in a scaled environment, such as using file replication or UNC share.

All content management servers use the Analytics index to search for contacts in the Experience Profile. To do this, the content management server(s) must have access to a search index – either over HTTP(S) for Solr, or on disk for Lucene. If you are running the xDB Cloud Edition this already includes an integrated indexing service for contact search.

Note

When you configure a dedicated processing server in a scaled environment, you must always use Solr. This is because Solr supports calls over HTTP(S), whereas calls to a Lucene index must be made from the same machine that hosts the index itself. Read the topic Using Solr or Lucene for more information.

Depending on whether you use Lucene or Solr for search and indexing there is a different set of configuration steps that you need to follow.

Note

In the `Sitecore.ContentSearch.Lucene.Index.Analytics.config` and `Sitecore.ContentSearch.Solr.Index.Analytics.config` files the `<ThrottlingEnabled>` setting is set to *True* by default. Change this setting to *False* to avoid a negative effect on performance during aggregation processing. This only applies to Sitecore 8.0. In later versions of Sitecore `<ThrottlingEnabled>` is set to *False* by default.

If you use Lucene follow the steps below and disable all Solr configuration files and if you use Solr follow the steps below and disable all Lucene configuration files.

To configure a processing server:

1. Install a standard Sitecore instance on the server you want to use for processing. For installation instructions, read the *Installing the Sitecore Experience Platform* guide on the Sitecore Developer Portal website.
2. Remove or restrict access to the client. You do not need the Sitecore client on a processing server.
3. In the *App_Config* folder, in the `ConnectionStrings.config` file, comment out the following connection strings:

   ```
   <add name="web" connectionString="….." />
   ```

   ```
   <add name="session" connectionString="mongodb:….."/>
   ```

4. Enable or disable configuration files for the:
   - Processing server
   - Lucene provider
   - Solr provider
5. Change the configuration settings for the:
   - Processing server
   - Solr provider

Important

When you configure xDB, you must synchronize all the servers in your solution to a single reliable time source, for example by means of the Network Time Protocol (NTP).

The aggregation of engagement automation states depends on the system time and changing this can lead to incorrect aggregation results or loss of data.

Files to enable or disable to configure a processing server

To configure a processing server, ensure that the following configuration files or .dll files are either enabled or disabled.

Note

Only files that are need to be changed from the default configuration are listed here.

Configuration file	Folder	Enable	Disable
Marketing platform			

Sitecore.Analytics.Automation.TimeoutProcessing.config	App_Config/Include		✓
Sitecore.Analytics.Tracking.config	App_Config/Include		✓
Sitecore.Analytics.Tracking.Database.config	App_Config/Include		✓
Sitecore.Analytics.Tracking.RobotDetection.config	App_Config/Include		✓
Sitecore.Marketing.Client.config	App_Config/Include		✓
Sitecore.WebDAV.config	App_Config/Include		✓
Sitecore.Xdb.Remote.Server.config	App_Config/Include	✓	
Sitecore.Xdb.Remote.Server.MarketingAssets.config	App_Config/Include	✓	
Path Analyzer			
Sitecore.PathAnalyzer.Client.config	App_Config/Include		✓
Sitecore.PathAnalyzer.RemoteClient.config	App_Config/Include		✓
Sitecore.PathAnalyzer.Services.config	App_Config/Include		✓
Sitecore.PathAnalyzer.Services.RemoteServer.config	App_Config/Include		✓
Sitecore.PathAnalyzer.StorageProviders.config	App_Config/Include		✓
Sitecore.PathAnalyzer.Client.dll	Website/bin		✓
Sitecore.PathAnalyzer.Services.dll	Website/bin		✓
Experience Analytics			
Sitecore.ExperienceAnalytics.Client.config	App_Config/Include/ExperienceAnalytics		✓
Sitecore.ExperienceAnalytics.Reduce.config	App_Config/Include/ExperienceAnalytics	✓	
Sitecore.ExperienceAnalytics.StorageProviders.config	App_Config/Include/ExperienceAnalytics		✓
Sitecore.ExperienceAnalytics.WebAPI.config	App_Config/Include/ExperienceAnalytics		✓
Experience Profile			
Sitecore.ExperienceProfile.config	App_Config/Include/ExperienceProfile		✓
Sitecore.ExperienceProfile.Client.config	App_Config/Include/ExperienceProfile		✓
Sitecore.ExperienceProfile.Reporting.config	App_Config/Include/ExperienceProfile		✓

Federated Experience Manager		
`Sitecore.FXM.Speak.config`	`App_Config/Include/FXM`	✓
`Sitecore.Services.Client.FXM.Enabler.config`	`App_Config/Include/FXM`	✓
List Management		
`Sitecore.ListManagement.Client.config`	`App_Config/Include/ListManagement`	✓
`Sitecore.ListManagement.config`	`App_Config/Include/ListManagement`	✓
`Sitecore.ListManagement.Services.config`	`App_Config/Include/ListManagement`	✓
Social Connected		
`Sitecore.Social.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ExperienceProfile.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.Facebook.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.GooglePlus.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.LinkedIn.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ProfileMapping.Facebook.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ProfileMapping.GooglePlus.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ProfileMapping.LinkedIn.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ProfileMapping.Twitter.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.SocialMarketer.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.Twitter.config`	`App_Config/Include/Social`	✓

Changes to configuration file settings

In the following configuration files, find the appropriate section and update each of the parameter values.

Note

In Sitecore, you can create your own names for databases and indexes. This table uses the default Sitecore names for all databases and indexes, such as *master*, *web* and *sitecore_marketing_asset_index_web*.

Configuration file	Section	Paramete
`Sitecore.Analytics.MarketingTaxonomy.config`	taxonomy/knownManagers/campaignGroupTaxonomyManager/<param desc="repository">	repositor
	taxonomy/knownManagers/channelTaxonomyManager/<param desc="repository">	repositor

	taxonomy/knownManagers/assetTaxonomyManager/<param desc="repository">	repositor
	taxonomy/customTaxonomyManager/<param desc="repository">	repositor
Sitecore.Buckets.config	configuration/sitecore/databases/ Comment out the web database section.	
Sitecore.ContentSearch.DefaultConfigurations.config	configuration/sitecore/contentSearch/ indexConfigurations/indexUpdateStrategies Comment out all strategies using the web database.	
Sitecore.Marketing.Config	marketingDefinitions/campaign/definitionManager/<param desc="repository">	repositor
	marketingDefinitions/goal/definitionManager/<param desc="repository">	repositor
	marketingDefinitions/outcome/definitionManager/<param desc="repository">	repositor
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.config	marketingDefinitions/asset/definitionManager	repositor
Sitecore.Xdb.Remote.Server.config	configuration/sitecore/customHandlers Comment out all handlers under custom handlers section.	
Sitecore.Xdb.Remote.Server.MarketingAssets.config	configuration/sitecore/customHandlers Comment out.	
web.config	configuration/sitecore<!-- DATABASES --> Comment out the web database section.	
	configuration/sitecore/<!-- SITES -->scheduling Comment out.	
	Configuration/sitecore/<!-- SITES --> Comment out any sites that use the web database.	

Files to enable or disable for Lucene

To configure a processing server when using Lucene for search and indexing, you need to enable or disable the following additional configuration files.

Note

All relevant Lucene and Solr configuration files are included in this section regardless of whether they are enabled or disabled by default.

To disable a configuration file change the extension from *.config* to *.disabled*.

Configuration file	Folder	Enable D
Marketing platform		
Sitecore.ContentSearch.Analytics.config	App_Config/Include	✓
Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config	App_Config/Include	

Enabled by default		✓	
`Sitecore.ContentSearch.Lucene.Index.Analytics.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Core.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Master.config` Enabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Lucene.Index.Web.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.DefaultIndexConfiguration.config` Disabled by default	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Analytics.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Core.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Master.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.Index.Web.config`	`App_Config/Include`		✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Master.config`	`App_Config/Include`		✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Web.config`	`App_Config/Include`		✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.IndexConfiguration.config`	`App_Config/Include`		✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Master.config` Disabled by default	`App_Config/Include`		✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Web.config` Disabled by default	`App_Config/Include`		✓
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include`		✓
Content Testing			
`Sitecore.ContentTesting.Lucene.IndexConfiguration.config` Enabled by default	`App_Config/Include/ContentTesting`	✓	
`Sitecore.ContentTesting.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/ContentTesting`		✓
Federated Experience Manager			
`Sitecore.FXM.Lucene.Index.DomainsSearch.config`	`App_Config/Include/FXM`		✓
`Sitecore.FXM.Solr.Index.DomainsSearch.config`	`App_Config/Include/FXM`		✓

Disabled by default			
List Management			
`Sitecore.ListManagement.Lucene.Index.List.config`	`App_Config/Include/ListManagement`		✓
`Sitecore.ListManagement.Lucene.IndexConfiguration.config`	`App_Config/Include/ListManagement`		✓
`Sitecore.ListManagement.Solr.Index.List.config` Disabled by default	`App_Config/Include/ListManagement`		✓
`Sitecore.ListManagement.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/ListManagement`		✓
Social Connected			
`Sitecore.Social.Lucene.Index.Master.config`	`App_Config/Include/Social`		✓
`Sitecore.Social.Lucene.Index.Web.config`	`App_Config/Include/Social`		✓
`Sitecore.Social.Lucene.IndexConfiguration.config`	`App_Config/Include/Social`		✓
`Sitecore.Social.Solr.Index.Master.config` Disabled by default	`App_Config/Include/Social`		✓
`Sitecore.Social.Solr.Index.Web.config` Disabled by default	`App_Config/Include/Social`		✓
`Sitecore.Social.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/Social`		✓

Files to enable or disable for Solr

To configure a processing server when using Solr for search and indexing, you need to enable or disable the following additional configuration files.

Note

All relevant Lucene and Solr configuration files are included in this section regardless of whether they are enabled or disabled by default.

To disable a configuration file change the extension from *.config* to *.disabled*.

Configuration file	Folder	Enable	D
Marketing platform			
`Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Analytics.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Core.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Master.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Web.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.DefaultIndexConfiguration.config`	`App_Config/Include`	✓	

`Sitecore.ContentSearch.Solr.Index.Analytics.config`	`App_Config/Include`	✓
`Sitecore.ContentSearch.Solr.Index.Core.config`	`App_Config/Include`	✓
`Sitecore.ContentSearch.Solr.Index.Master.config`	`App_Config/Include`	✓
`Sitecore.ContentSearch.Solr.Index.Web.config`	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Master.config`	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Web.config`	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.IndexConfiguration.config`	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Master.config` Disabled by default	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Web.config` Disabled by default	`App_Config/Include`	
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include`	
Content Testing		
`Sitecore.ContentTesting.Lucene.IndexConfiguration.config`	`App_Config/Include/ContentTesting`	
`Sitecore.ContentTesting.Solr.IndexConfiguration.config`	`App_Config/Include/ContentTesting`	✓
Federated Experience Manager		
`Sitecore.FXM.Lucene.Index.DomainsSearch.config`	`App_Config/Include/FXM`	
`Sitecore.FXM.Solr.Index.DomainsSearch.config` Disabled by default	`App_Config/Include/FXM`	
List Management		
`Sitecore.ListManagement.Lucene.Index.List.config`	`App_Config/Include/ListManagement`	
`Sitecore.ListManagement.Lucene.IndexConfiguration.config`	`App_Config/Include/ListManagement`	
`Sitecore.ListManagement.Solr.Index.List.config` Disabled by default	`App_Config/Include/ListManagement`	
`Sitecore.ListManagement.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/ListManagement`	
Social Connected		
`Sitecore.Social.Lucene.Index.Master.config`	`App_Config/Include/Social`	

Sitecore.Social.Lucene.Index.Web.config	App_Config/Include/Social
Sitecore.Social.Lucene.IndexConfiguration.config	App_Config/Include/Social
Sitecore.Social.Solr.Index.Master.config Disabled by default	App_Config/Include/Social
Sitecore.Social.Solr.Index.Web.config Disabled by default	App_Config/Include/Social
Sitecore.Social.Solr.IndexConfiguration.config Disabled by default	App_Config/Include/Social

Solr configuration

If you use Solr as your search and indexing provider, first ensure that you have enabled all Solr configuration files required on a processing server. If you use multiple cores, no further configuration is necessary.

It is no longer necessary to store each default index in a separate Solr core. However, for optimal performance this approach is preferred. If you store each default index in a separate Solr core the names of these cores will match the names of the Sitecore indexes.

Solr cores required on a processing server:

- sitecore_analytics_index
- sitecore_testing_index
- sitecore_suggested_test_index

Note

Multiple Solr cores may improve the performance of your search and indexing. For example, you should use separate cores for each index in a production environment and only share indexes on a single core in a development environment. The Solr website provides more detailed guidance and advice when configuring Solr.

Renaming Solr cores

You can change the default Solr core names to your own custom names and can also store several Sitecore indexes in a shared Solr core.

If you want to change the default Solr core names and use your own custom names, then you also need to change the names in each Sitecore Solr configuration file.

To set a Solr core for a Sitecore index:

1. In the App_Config\Include folder, open the configuration file where the Sitecore index is declared. For example, in the Sitecore.ContentSearch.Solr.Index.Core.config file.
2. Navigate to the section where the index is declared: configuration/sitecore/contentSearch/configuration/indexes/:

```
<index id="..." type="Sitecore.ContentSearch.SolrProvider.SolrSearchIndex,
 Sitecore.ContentSearch.SolrProvider">
   <param desc="name">$(id)</param>
   <param desc="core">$(id)</param>
```

3. Change value of the core parameter to the custom name of the Solr core you want to use.

   Follow the same procedure for each Solr configuration file where you want to change the Solr core name.

Configuring aggregation agents or threads

To configure aggregation agents or threads on a processing server:

1. Open the Sitecore.Analytics.Processing.Aggregation.Services.config file in an XML editor. You can specify how many aggregation agents you want to run at the same time. Start by using the default settings.
2. Edit the <MaxThreads> setting to specify the number of aggregation agents (threads) that you want to run on the server. You can also specify how many cleanup and recovery threads that you need. Adjust these settings depending on your available hardware and business requirements.

```
<!-- Aggregation Module: -->
<module type="Sitecore.Analytics.Aggregation.AggregationModule"
   singleInstance="true">
  <BackgroundServices hint="list:Add">
    <aggregator type="Sitecore.Analytics.Aggregation.BackgroundService">
      <param desc="agentName">aggregation/aggregator</param>
      <Interval>0.00:00:15</Interval>
      <MaxThreads>16</MaxThreads>
    </aggregator>
    <cleanup type="Sitecore.Analytics.Aggregation.BackgroundService">
```

```
        <param desc="agentName">aggregation/cleanup</param>
        <Interval>0.00:00:15</Interval>
        <MaxThreads>16</MaxThreads>
      </cleanup>
      <recovery type="Sitecore.Analytics.Aggregation.BackgroundService">
        <param desc="agentName">aggregation/recovery</param>
        <Interval>0.00:00:15</Interval>
        <MaxThreads>16</MaxThreads>
      </recovery>
      <history type="Sitecore.Analytics.Aggregation.BackgroundService">
        <param desc="agentName">aggregation/historyWorker</param>
        <Interval>0.00:00:15</Interval>
        <MaxThreads>16</MaxThreads>
      </history>
      <historyCompletionCheck
       type="Sitecore.Analytics.Aggregation.BackgroundService">
        <param desc="agentName">aggregation/historyCompletionCheck</param>
        <Interval>0.00:00:15</Interval>
        <MaxThreads>16</MaxThreads>
      </historyCompletionCheck>
    </BackgroundServices>
  </module>
```

Use the default values and then over time adjust these settings to optimize your aggregation server or servers.

3. Adjust the `Max Pool Size` property for the reporting database. The pool capacity should be at least about 120% of the number of agents (including processing, history, automation workers and clean-up) that you have configured. This setting ensures that each thread has its own dedicated connection and will not have to wait for another thread to release its connection before running.
4. To set the pool size, open the reporting database connection string configuration file, add the `Max Pool Size` property and set an appropriate value.

In this type of configuration, the processing server performs two functions: aggregation and processing. These are the two main roles for a processing server.

Configuring the Path Analyzer on a remote processing/reporting shared server

In this type of configuration, the content management server does not have direct access to the `Sitecore.Analytics` SQL database. Instead, the *Path Analyzer* client retrieves data from a remote server that does have direct access to the SQL database using Web API services.

If you have a remote processing environment combined with a reporting server then make the following configuration changes to that environment.

The following configuration files should be enabled:

- `Sitecore.PathAnalyzer.Services.RemoteServer.config`
- `Sitecore.PathAnalyzer.Services.config`
- `Sitecore.PathAnalyzer.Processing.config`

The following configuration files should be disabled:

- `App_Config/Include/Sitecore.PathAnalyzer.Client.config`
- `App_Config/Include/Sitecore.PathAnalyzer.RemoteClient.config`

The following assemblies should be disabled:

- `bin/Sitecore.PathAnalyzer.Client.dll`

Note

If you have a dedicated Reporting server instance, the Processing server instance should not have the following configuration files enabled: `Sitecore.PathAnalyzer.Services.RemoteServer.config` and `Sitecore.PathAnalyzer.Services.config`.

The *Services* configuration files should only be enabled on servers that need to expose the Path Analyzer Web API. The remote server config should only be enabled on servers that need to expose the Path Analyzer Web API to external clients.

Send feedback about the documentation to docsite@sitecore.net.

Configure a reporting database server

You can configure a Sitecore reporting database (SQL Server) server to run on a separate SQL Server instance, or together on the same server with other Sitecore applications.

Creating a separate server for the reporting database gives you more control over performance and scalability.

Note

As the reporting database only requires SQL Server, there is no need to install a Sitecore instance when you create a reporting database server.

To configure a Microsoft SQL Server instance as a dedicated reporting server:

1. Install SQL Server 2008 R2 SP1 or later.
2. Configure SQL Server according to the system requirements and Sitecore CMS installation instructions.
3. If you have already configured separate content management and processing servers, open the `ConnectionStrings.config` file and modify the connection strings to point to your reporting database server.

   Note

   Ensure that you configure connection strings for any other types of servers that need to connect to the reporting database server.

4. Test your connections to ensure that your SQL Server instance is configured and running correctly. Configure connection strings according to the Sitecore CMS installation instructions.

Note

You can only scale a SQL Server reporting database server vertically by adding more resources to a single server.

Send feedback about the documentation to docsite@sitecore.net.

Configure a Reporting Service server

The Sitecore Reporting Service fetches reporting data from various data sources (for example, the collection or reporting databases) to use in Sitecore reporting applications, such as Experience Analytics. The Reporting Service can run on the same server alongside other server features, such as processing, or can run independently on a dedicated server.

In the reporting architecture, the Reporting Service sits between Sitecore reporting applications and the rest of the Sitecore Experience Database (xDB) architecture and handles the processing of all your report queries.

Before you start configuring server roles ensure that you have read the hardware guidelines for the Sitecore Experience Platform (XP), and the Sitecore Experience Database (xDB).

Note

The information in this topic is valid for Sitecore Experience Platform 8.0 Update 3.

In this type of configuration, you use the Reporting Service server with the Reporting Service server feature.

This topic includes the following:

- Files to enable or disable to configure a Reporting Service server
- Changes to configuration file settings
- Files to enable or disable for Lucene
- Files to enable or disable for Solr
- Solr configuration
- Enabling communication with the Remote Reporting Service

Search and indexing considerations

Decide which search and indexing provider you want to use. You can choose Lucene, Solr or a combination of the two depending on the business purpose of your website and the type of server architecture you want to implement.

Depending on whether you use Lucene or Solr for search and indexing there is a different set of configuration steps that you need to follow. Before you choose a search provider, first read more about Using Solr or Lucene with Sitecore, as you have several different options.

If you use Lucene follow the steps below and disable all Solr configuration files and if you use Solr follow the steps below and disable all Lucene configuration files.

To configure a dedicated Reporting Service server:

1. Install a standard Sitecore instance on the server you want to use for the Reporting Service. For installation instructions, read the *Installing the Sitecore Experience Platform* guide on the Sitecore Developer Portal website.
2. Remove or restrict access to the client. You do not need the Sitecore client on a Reporting Service server.
3. In the `/App_Config/ConnectionStrings.config` file, comment out the following connection strings:

   ```
   <add name="tracking.live " connectionString="mongodb:….."/>
   
   <add name="tracking.history" connectionString="mongodb:….."/>
   
   <add name="tracking.contact" connectionString="mongodb:….."/>
   ```

4. Enable or disable configuration files for the:
   - Reporting Service server
   - Lucene provider
   - Solr provider
5. Change the configuration settings for the:
   - Reporting Service server
   - Solr provider

Important

When you configure xDB, you must synchronize all the servers in your solution to a single reliable time source, for example by means of the Network Time Protocol (NTP).

The aggregation of engagement automation states depends on the system time and changing this can lead to incorrect aggregation results or loss of data.

Files to enable or disable to configure a Reporting Service server

To configure a Reporting Service server, ensure that the following configuration files or .dll files are either enabled or disabled.

Note

Only files that are need to be changed from the default configuration are listed here.

Note

If you configure a combined processing/reporting server then you need to follow two sets of configuration instructions. Some configuration files are enabled by default, so if you first configure the processing feature, when you later configure the reporting feature you will notice that you have already changed the default configuration.

Configuration file	Folder	Enable	Disable
Marketing platform			
Sitecore.Analytics.Automation.TimeoutProcessing	App_Config/Include		✓
Sitecore.Analytics.Processing.Aggregation.Services.config	App_Config/Include		✓
Sitecore.Analytics.Processing.Services.config	App_Config/Include		✓
Sitecore.Analytics.Tracking.Aggregation.config	App_Config/Include		✓
Sitecore.Analytics.Tracking.config	App_Config/Include		✓
Sitecore.Analytics.Tracking.Database.config	App_Config/Include		✓
Sitecore.Analytics.Tracking.RobotDetection.config	App_Config/Include		✓
Sitecore.Marketing.Client.config	App_Config/Include		✓
Sitecore.Processing.config	App_Config/Include		✓
Sitecore.WebDAV.config	App_Config/Include		✓
Sitecore.Xdb.Remote.Server	App_Config/Include	✓	
Path Analyzer			
Sitecore.PathAnalyzer.Client.config	App_Config/Include		✓
Sitecore.PathAnalyzer.Processing.config	App_Config/Include		✓
Sitecore.PathAnalyzer.RemoteClient.config	App_Config/Include		✓
Sitecore.PathAnalyzer.Services.RemoteServer.config	App_Config/Include	✓	
Content Testing			
Sitecore.ContentTesting.Processing.Aggregation.config	App_Config/Include/Content Testing		✓
Experience Analytics			
Sitecore.ExperienceAnalytics.Aggregation.config	App_Config/Include/ExperienceAnalytics		✓
Sitecore.ExperienceAnalytics.Client.config	App_Config/Include/ExperienceAnalytics		

		✓
`Sitecore.ExperienceAnalytics.Reduce.config`	`App_Config/Include/ExperienceAnalytics`	✓
`Sitecore.ExperienceAnalytics.WebAPI.config`	`App_Config/Include/ExperienceAnalytics`	✓
Experience Profile		
`Sitecore.ExperienceProfile.Client.config`	`App_Config/Include/ExperienceProfile`	✓
Federated Experience Manager		
`Sitecore.FXM.Speak.config`	`App_Config/Include/FXM`	✓
`Sitecore.Services.Client.FXM.Enabler.config`	`App_Config/Include/FXM`	✓
List Management		
`Sitecore.ListManagement.Client.config`	`App_Config/Include/ListManagement`	✓
`Sitecore.ListManagement.config`	`App_Config/Include/ListManagement`	✓
`Sitecore.ListManagement.Services.config`	`App_Config/Include/ListManagement`	✓
Social Connected		
`Sitecore.Social.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ExperienceProfile.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.Facebook.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.GooglePlus.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.LinkedIn.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ProfileMapping.Facebook.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ProfileMapping.GooglePlus.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ProfileMapping.LinkedIn.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.ProfileMapping.Twitter.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.SocialMarketer.config`	`App_Config/Include/Social`	✓
`Sitecore.Social.Twitter.config`	`App_Config/Include/Social`	✓

Changes to configuration file settings

In the following configuration files, find the appropriate section and update each of the parameter values.

Note

In Sitecore, you can create your own names for databases and indexes. This table uses the default Sitecore names for all databases and indexes, such as *master*, *web* and *sitecore_marketing_asset_index_web*.

Configuration file	Section	Paramet
Sitecore.Analytics.MarketingTaxonomy.config	taxonomy/knownManagers/campaignGroupTaxonomyManager/<param desc="repository">	ref
	taxonomy/knownManagers/channelTaxonomyManager/<param desc="repository">	repositor
	taxonomy/knownManagers/assetTaxonomyManager/<param desc="repository">	repositor
	taxonomy/knownManagers/customTaxonomyManager/<param desc="repository">	repositor
Sitecore.Marketing.Config	marketingDefinitions/campaign/definitionManager/<param desc="repository">	ref
	marketingDefinitions/goal/definitionManager/<param desc="repository">	repositor
	marketingDefinitions/outcome/definitionManager/<param desc="repository">	repositor
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.config	marketingDefinitions/asset/definitionManager/<param desc="repository">	ref
	marketingDefinitions/asset/repositories/item/	databaseN
	marketingDefinitions/asset/repositories/item/	indexName

Files to enable or disable for Lucene

To configure a Reporting Service server when using Lucene for search and indexing, you need to enable or disable the following additional configuration files.

Note

All relevant Lucene and Solr configuration files are included in this section regardless of whether they are enabled or disabled by default.

To disable a configuration file change the extension from *.config* to *.disabled*.

Configuration file	Folder	Enable D
Marketing platform		
Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config Enabled by default	App_Config/Include	✓
Sitecore.ContentSearch.Lucene.Index.Analytics.config Enabled by default	App_Config/Include	✓
Sitecore.ContentSearch.Lucene.Index.Core.config Enabled by default	App_Config/Include	✓
Sitecore.ContentSearch.Lucene.Index.Master.config Enabled by default	App_Config/Include	✓
Sitecore.ContentSearch.Lucene.Index.Web.config Enabled by default	App_Config/Include	✓
Sitecore.ContentSearch.Solr.DefaultIndexConfiguration.config	App_Config/Include	

Disabled by default	
`Sitecore.ContentSearch.Solr.Index.Analytics.config` Disabled by default	`App_Config/Include`
`Sitecore.ContentSearch.Solr.Index.Core.config` Disabled by default	`App_Config/Include`
`Sitecore.ContentSearch.Solr.Index.Master.config` Disabled by default	`App_Config/Include`
`Sitecore.ContentSearch.Solr.Index.Web.config` Disabled by default	`App_Config/Include`
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Master.config`	`App_Config/Include`
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Web.config`	`App_Config/Include`
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.IndexConfiguration.config`	`App_Config/Include`
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Master.config` Disabled by default	`App_Config/Include`
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Web.config` Disabled by default	`App_Config/Include`
`Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include`
Content Testing	
`Sitecore.ContentTesting.Lucene.IndexConfiguration.config`	`App_Config/Include/ContentTesting`
`Sitecore.ContentTesting.Solr.IndexConfiguration.config` *Disabled by default	`App_Config/Include/ContentTesting`
Federated Experience Manager	
`Sitecore.FXM.Lucene.Index.DomainsSearch.config` Enabled by default	`App_Config/Include/FXM`
`Sitecore.FXM.Solr.Index.DomainsSearch.config`	`App_Config/Include/FXM`
List Management	
`Sitecore.ListManagement.Lucene.Index.List.config`	`App_Config/Include/ListManagement`
`Sitecore.ListManagement.Lucene.IndexConfiguration.config`	`App_Config/Include/ListManagement`
`Sitecore.ListManagement.Solr.Index.List.config` Disabled by default	`App_Config/Include/ListManagement`
`Sitecore.ListManagement.Solr.IndexConfiguration.config`	`App_Config/Include/ListManagement`

Disabled by default			
Social Connected			
`Sitecore.Social.Lucene.Index.Master.config`	`App_Config/Include/Social`		✓
`Sitecore.Social.Lucene.Index.Web.config`	`App_Config/Include/Social`		✓
`Sitecore.Social.Lucene.IndexConfiguration.config`	`App_Config/Include/Social`		✓
`Sitecore.Social.Solr.Index.Master.config` Disabled by default	`App_Config/Include/Social`		✓
`Sitecore.Social.Solr.Index.Web.config` Disabled by default	`App_Config/Include/Social`		✓
`Sitecore.Social.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/Social`		✓

Files to enable or disable for Solr

To configure a content delivery server when using Solr for search and indexing, you need to enable or disable the following additional configuration files.

Note

All relevant Lucene and Solr configuration files are included in this section regardless of whether they are enabled or disabled by default.

To disable a configuration file change the extension from *.config* to *.disabled*.

Configuration file	Folder	Enable D	
Marketing platform			
`Sitecore.ContentSearch.Lucene.DefaultIndexConfiguration.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Analytics.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Core.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Master.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Lucene.Index.Web.config`	`App_Config/Include`		✓
`Sitecore.ContentSearch.Solr.DefaultIndexConfiguration.config`	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Analytics.config`	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Analytics.config` Disabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Core.config` Disabled by default	`App_Config/Include`	✓	
`Sitecore.ContentSearch.Solr.Index.Master.config` Disabled by default	`App_Config/Include`	✓	

Sitecore.ContentSearch.Solr.Index.Web.config Disabled by default	App_Config/Include	✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Master.config	App_Config/Include	✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.Index.Web.config	App_Config/Include	✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Lucene.IndexConfiguration.config	App_Config/Include	✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Master.config Disabled by default	App_Config/Include	✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.Index.Web.config Disabled by default	App_Config/Include	✓
Sitecore.Marketing.Definitions.MarketingAssets.Repositories.Solr.IndexConfiguration.config Disabled by default	App_Config/Include	✓
Content Testing		
Sitecore.ContentTesting.Lucene.IndexConfiguration.config	App_Config/Include/ContentTesting	✓
Sitecore.ContentTesting.Solr.IndexConfiguration.config *Disabled by default	App_Config/Include/ContentTesting	✓
Federated Experience Manager		
Sitecore.FXM.Lucene.Index.DomainsSearch.config Enabled by default	App_Config/Include/FXM	✓
Sitecore.FXM.Solr.Index.DomainsSearch.config	App_Config/Include/FXM	✓
List Management		
Sitecore.ListManagement.Lucene.Index.List.config	App_Config/Include/ListManagement	✓
Sitecore.ListManagement.Lucene.IndexConfiguration.config	App_Config/Include/ListManagement	✓
Sitecore.ListManagement.Solr.Index.List.config Disabled by default	App_Config/Include/ListManagement	✓
Sitecore.ListManagement.Solr.IndexConfiguration.config Disabled by default	App_Config/Include/ListManagement	✓
Social Connected		
Sitecore.Social.Lucene.Index.Master.config	App_Config/Include/Social	✓
Sitecore.Social.Lucene.Index.Web.config	App_Config/Include/Social	✓
Sitecore.Social.Lucene.IndexConfiguration.config	App_Config/Include/Social	✓
Sitecore.Social.Solr.Index.Master.config	App_Config/Include/Social	✓

Disabled by default		
`Sitecore.Social.Solr.Index.Web.config` Disabled by default	`App_Config/Include/Social`	✔
`Sitecore.Social.Solr.IndexConfiguration.config` Disabled by default	`App_Config/Include/Social`	✔

Solr configuration

If you use Solr as your search and indexing provider, first ensure that you have enabled all Solr configuration files required on a Reporting Service server. If you use multiple cores, no further configuration is necessary.

It is no longer necessary to store each default index in a separate Solr core. However, for optimal performance this approach is preferred. If you store each default index in a separate Solr core the names of these cores will match the names of the Sitecore indexes.

Solr cores required on a Reporting Service server:

- `sitecore_analytics_index`

Note

Multiple Solr cores may improve the performance of your search and indexing. For example, you should use separate cores for each index in a production environment and only share indexes on a single core in a development environment. The Solr website provides more detailed guidance and advice when configuring Solr.

Renaming Solr cores

You can change the default Solr core names to use your own custom names and can also store several Sitecore indexes in a shared Solr core.

If you want to change the default Solr core names and use your own custom names, then you need to change the names in each Sitecore Solr configuration file.

To set a Solr core for a Sitecore index:

1. In the `App_Config\Include` folder, open the configuration file where the Sitecore index is declared. For example, in the `Sitecore.ContentSearch.Solr.Index.Core.config` file.
2. Navigate to the section where the index is declared: `configuration/sitecore/contentSearch/configuration/indexes/`:

```
<index id="..." type="Sitecore.ContentSearch.SolrProvider.SolrSearchIndex,
 Sitecore.ContentSearch.SolrProvider">
   <param desc="name">$(id)</param>
   <param desc="core">$(id)</param>
```

3. Change value of the `core` parameter to the custom name of the Solr core you want to use.

   Follow the same procedure for each Solr configuration file where you want to change the Solr core name.

Enabling communication with the Remote Reporting Service

To enable communication between a content management server (client) and the Remote Reporting Service server there are some additional steps you need to follow.

Note

These steps do not apply to Sitecore xDB Cloud Edition.

To configure a content management server and a remote Reporting Service server:

1. On your content management server client, in the *App_ Config* folder, enable the following configuration files by removing `.disabled` from the file name:
   - `Sitecore.Xdb.Remote.Client.config`
   - `Sitecore.Xdb.Remote.Client.MarketingAssets.config`
   - `Sitecore.PathAnalyzer.RemoteClient.config`
2. Under the `<httpTransportFactory>` node, change the default value of the desc parameter to the hostname of your Reporting Service server instance.

   For example:

```
<httpTransportFactory
       type="Sitecore.Analytics.Commons.ConfigurationBasedHttpTransportFactory,
       Sitecore.Analytics" singleInstance="true">
    <param desc="serviceUrl">http://reportingservice/</param>
</httpTransportFactory>
```

Note

Ensure that the *Sitecore.Xdb.Remote.Server.config*, *Sitecore.Xdb.Remote.Server.MarketingAssets.config*, *Sitecore.PathAnalyzer.Services.RemoteServer.config* files are enabled on your remote *Reporting Service* server.

Configure multiple managed websites

A single Sitecore instance uses multiple websites to manage content delivery, content management, and many other features. However, by default there is only one published website. This website corresponds to the `<site name="website"…/>` definition in the `Sitecore.config` file.

You can configure additional managed websites for different purposes and different domains by adding them to the site definition node in the `Sitecore.config` file. For example, you can define a new site to be your published public-facing site or to be used by content editors to access the content management system.

To configure an extra Sitecore website:

1. [Add a patch file](#) containing the new site definition.

   Note

   You must not add the definition directly to the `<sites>` list in the `Sitecore.config` file because changes to the file can be lost during an upgrade process.

   The following example adds a website called *admin*:

   ```
   <?xml version="1.0"?>
   <configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
       <sitecore>
           <sites>
             <site name="mysite" patch:after="site[@name='modules_website']"
               targetHostName="rhino.acme.com”
               enableTracking="true" virtualFolder="/" physicalFolder="/"
               rootPath="/sitecore/content/mysite"
               startItem="/home" database="web" domain="extranet"
               allowDebug="true" cacheHtml="true" htmlCacheSize="50MB" registryCacheSize="0"
               viewStateCacheSize="0" xslCacheSize="25MB" filteredItemsCacheSize="10MB"
               enablePreview="true" enableWebEdit="true" enableDebugger="true"
               disableClientData="false" cacheRenderingParameters="true"
               renderingParametersCacheSize="10MB" />
           </sites>
       </sitecore>
   </configuration>
   ```

2. For each added website, configure the website attributes depending on the purpose of the website. The attributes and their purpose are described in the comments before the `<site>` section in the `Sitecore.config` file.

   Note

   You must define all attributes for each added website. Attributes can vary from version to version due to updates or new features in the product. Therefore, after any upgrade you must check the attributes for the default site in the `Sitecore.config` file, because there can be changes that require you to update the attributes of any added websites.

3. Enter the host name of the website in the `hosts` file of your computer or server, in the following format:

   ```
   102.54.94.97     rhino.acme.com          # source server
   ```

Website context

Sitecore determines the website context based on the incoming URL and the following two attributes:

- `hostName` – Specifies the name of the website as it appears in the incoming URL. It can contain wildcards.
- `virtualFolder` – The physical path to the website root folder.

When an incoming URL is received, Sitecore first compares it to the `hostName` attribute of each site in the `<sites>` list.

If the `hostName` attribute matches the incoming URL or the `hostName` attribute is empty, Sitecore compares the `virtualFolder` attribute next.

Sitecore uses the first site definition in the `<sites>` list where the `hostName` and `virtualFolder` attributes match the incoming URL as the context site.

Note

Sitecore checks for sites in the order in which they are included in the list. Therefore, sites must be ordered starting with the most specific entry and ending with the most general entry. For example, if you have a website where `hostName` is mysite.com.au and one where `hostName` is mysite.com, the entry for mysite.com.au must be first in the list. Otherwise, an incoming URL for mysite.com.au will match the entry for mysite.com first.

Send feedback about the documentation to docsite@sitecore.net.

Configuring servers

Just like its predecessor, the Sitecore Experience Platform supports a standalone, all-in-one configuration, retaining the ability to run on a single server. This enables easy deployment of a development environment or a small, low-traffic production instance. However, you can see the real benefits and flexibility of the xDB architecture when you start trying to improve the performance of your application.

To achieve better performance and scalability, you can configure dedicated servers for different purposes, such as for content delivery or data processing. You can choose from a list of server features that you can configure on a single server or on multiple dedicated servers.

Configuration

In practical terms, to assign features to a dedicated server, you simply configure a reduced set of components on a server. You can do this by disabling or enabling configuration files in your Sitecore installation. This gives you flexibility and fine-tuned control over the tasks and activities that you assign to each of your Sitecore instances.


Picture 2

Benefits

By dividing your Sitecore application up into multiple servers performing different activities, you increase your scaling options. For improved application performance, you can scale both vertically (by scaling up the hardware on each server) and horizontally (by adding more servers to perform each activity). For improved availability and reliability, you can create clusters, such as for content delivery or data processing.

For additional flexibility, you can also configure a single server to perform multiple server tasks. For example, you could enable content management, content delivery, and reporting on a single server but move the MongoDB collection database to another server.

You should configure dedicated servers for different purposes:

- Content delivery server
- Content management server
- Processing/aggregation server
- Reporting Service server
- Collection database server
- Reporting database server
- Session database server

Send feedback about the documentation to docsite@sitecore.net.

Database connection strings for configuring servers

In the Sitecore Experience Platform (XP), when you configure servers to perform different roles in a scalable environment, you must configure several database connection strings for each server type.

To configure the connection strings for a Sitecore server instance, edit the `ConnectionStrings.config` file located in the `Website\App_Config` folder.

When you configure a Sitecore server, enable or disable the following connection strings:

Database	Connection String	CD server	CM server	Processing server	Reporting Service server
SQL Server	`core`	Enable	Enable	Enable	Enable
	`master`	Disable	Enable	Enable	Enable
	`web`	Enable	Enable	Disable	Enable
	`reporting`	Disable	Enable	Enable	Enable
	`reporting.secondary` (optional)	NA	NA	Enable	Enable

MongoDB	`analytics`	Enable	Enable	Enable	Enable
	`tracking.live`	Enable	Enable	Enable	Enable
	`tracking.history`	Disable	Disable	Enable	Enable
	`tracking.contact`	Enable	Enable	Enable	Disable

Notes:

- Content delivery servers do not require all the connection strings to be active.

  For example, there are some security benefits in removing the reporting connection string. Either remove these connection strings completely or add comment tags to hide them.

- To perform historical processing by rebuilding the reporting database, you must add and enable an additional SQL Server connection string: `reporting.secondary`.

Send feedback about the documentation to docsite@sitecore.net.

Live session agent configuration settings

The Live Session Agent module enables you to trigger and process engagement automation states (EAS states) by timeout closer to the timeout delay set on the automation state.

Note

Sitecore 8.0 and 8.1 supports *Sitecore Engagement Automation Live Session Agent 1.0*. which is available to download from the Sitecore Developer Portal.

After you have installed the module, you should test the performance of your Sitecore solution so that you can then fine-tune the configuration settings of the module. When you test, you should note the following metrics:

- System throughput
- Request latency
- Load on the session state database – measured in transactions per second or the equivalent, both when the system is in an idle state and when loaded.
- CPU usage on content delivery and processing servers – both when idle and when the system is loaded.
- The average processing delay on EAS states (performance counter in the component).

You can change the following configuration file settings to fine-tune the performance of the Live Session Agent. The configuration files are stored in the Include folder (`Website\App_Config\Include\`).

Processing settings

Use the `Sitecore.EngagementAutomation.LiveSessionAgent.Processing.config` configuration file to configure Live Session Agent processing settings.

Wake-up interval of EAS Live session

The wake-up interval is the setting that controls how frequently the Live Session Agent polls the queue of contact sessions for new EAS items to be processed.

Setting	Description	Config sample

`Interval`	Specify the interval between each EAS state item that is processed.	`<liveSessionAgent type="Sitecore.Analytics.Core.BackgroundService">`
		`<param desc="agentName">automation/liveSession/agent</param>`
		`<Interval>0.00:00:15</Interval>`

The default interval value is 15 seconds. If you set this parameter too high, it can create a significant delay between EAS states becoming due and actually getting processed. Setting this parameter to a low value imposes a sustained load on the session state database (MongoDB or SQL Server) even when the system is idle.

Maximum number of parallel threads

This setting enables you to set the number of parallel threads for processing contacts with pending automation states. This setting is specified in the following node: `sitecore\automation\livesession\agent\MaxDegreeOfParallelism`

Setting	Description	Sample

`MaxDegreeOfParallelism`	Specify the maximum number of parallel threads that you want to allocate for processing.	`<agent type="Sitecore.EngagementAutomation.LiveSessionAgent.Agent">`
		`<MaxDegreeOfParallelism>4</MaxDegreeOfParallelism>`
		`</agent>`

The Live Session agent allocates processing threads from the same thread pool as ASP.NET worker threads, which may have an impact on your overall system throughput. You should not set the number of threads to more than 20% of the number of ASP.NET worker threads on a content delivery server. On a dedicated processing server, the number of threads can reach the size of the .NET thread pool if necessary.

Live session agent settings

Use the `Sitecore.EngagementAutomation.LiveSessionAgent.config` configuration file to configure general settings for the Live Session Agent.

Number of slots

This setting determines the number of slots that store queued contact identifiers for processing.

Setting	Description	Sample

`Number of Slots`	Specify the number of slots that you want to make available for storing queued contact identifiers for processing.	`<slotAllocationStrategy type="Sitecore.EngagementAutomation.LiveSessionAgent.Processing.SlotAllocationStrategy">`
		`  <!-- Number of slots stored in contact (shared) session storing chunks of queue of contact live sessions scheduled`
		`  -->`
		`  <param desc="Number of Slots">100</param>`
		`</slotAllocationStrategy>`

The number of slots you choose depends on the number of concurrent requests supported by the system. The default value is 100. This means that if your system has a load of 200 RPS, each slot will be locked and updated approximately 2 times per second.

Slots are stored in a shared session as session state items. Setting the number of slots to a low value will directly affect request latency and throughput of the system. Setting the number of slots to a high value can apply a sustained load on the session database even when the system is idle. In addition, this may cause unnecessary delays in processing EAS states, because more slots need to be scanned to find pending contact sessions.

Maximum number of items to lock for processing

This setting determines the number of items taken from each slot for processing during agent iterations.

Setting	Description	Sample

`MaxAmountOfItemsToLockForProcessing`	Specify the maximum number of items you want to lock for processing	`<queue type="Sitecore.EngagementAutomation.LiveSessionAgent.Processing.ProcessingTimeout`
		`  …`
		`  <!-- Maximum amount of items taken from each slot to return for processing as expired.`
		`  -->`
		`  <MaxAmountOfItemsToLockForProcessing>8</MaxAmountOfItemsToLockForProcessing>`
		`</queue>`

To eliminate possible significant delays when processing EAS timeouts, the system tries to access all existing slots on each polling interval and takes only a portion of data from each slot for processing. Setting this parameter to a low value can cause high traffic to the session database when the Live Session Agent actively processes EAS states. Setting this parameter to a high value can cause unnecessary delays between EAS states becoming due and getting processed.

Note

To calculate the value, multiply the number of contact sessions by the processing interval and divide that by the number of available slots.

Send feedback about the documentation to docsite@sitecore.net.

Using patch files to customize the Sitecore configuration

You use patch files to add or change configuration settings in Sitecore. Sitecore merges the patch files with the `Sitecore.config` file to create the configuration file that is used at runtime.

This topic describes:

- Patch file placement
- Patch file syntax
- Previewing the result of patching

Patch file placement

You must place patch files in the `/App_Config/Include` folder of the Sitecore installation, either directly or in a subfolder. You must give the patch files names ending with the extension `.config`.

Sitecore merges patch files in alphabetical order. Files within a folder are merged before files in subfolders within the folder. For example, the file `/App_Config/Include/mywebsite9.config` is merged before the file `/App_Config/Include/Myfolder/mywebsite1.config`.

If Sitecore finds the same configuration in multiple patch files, the last patch file to be merged overwrites the previous versions of the configuration.

For example, if all the files in the following list define the same configuration, Sitecore only uses the configuration from the last file listed:

- `/App_Config/Include/z.config`
- `/App_Config/Include/b/b.config`

- `/App_Config/Include/b/c.config`
- `/App_Config/Include/c/a.config` [used]

Patch file syntax

The following XML namespaces contain all the attributes and elements related to patching:

- Patch – http://www.sitecore.net/xmlconfig/
- Set – http://www.sitecore.net/xmlconfig/set/

You must declare the namespaces before you can use them in a patch file. Normally, you declare them at the beginning of the file:

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/"

xmlns:set="http://www.sitecore.net/xmlconfig/set/">

. . . . .

. . . . .

</configuration>
```

When Sitecore applies the changes from a patch file, it tries to match each element in the patch file to an element in the existing configuration. It matches elements using the combination of the element name and all the attributes of the element. If there is a match, Sitecore updates the existing element. If there is no match, Sitecore inserts a new element.

Note

If a patch file element matches more than one element in the existing configuration, Sitecore applies the update to the first matching element that it finds. It is therefore strongly recommended that you specify enough attributes of the element to uniquely identify it.

By default, Sitecore inserts new elements at the end of the configuration section that the elements belong to. You can use a `patch:` attribute in an element to specify where it is placed relative to existing elements, or to specify that it replaces, updates, or deletes an existing element of the same name. You can use the following attributes:

- `patch:before` – inserts the element before the specified element.
- `patch:after` – inserts the element after the specified element.
- `patch:attribute` – defines or replaces the specified attribute of the element. You can achieve the same effect using the `set` namespace.

For example, to add a site before the site with the name `website`, use the following syntax:

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">

  <sitecore>

    <sites>

      <site patch:before="*[@name='website']" name="mysite" ... />

    </sites>

  </sitecore>

</configuration>
```

You can use examples of patch files to help you change the Sitecore configuration.

Note

In the `/App_Config/Include` folder, there are several files ending with the extension `.example`. In these files, you can see examples of how to patch configuration elements. To include an example file in the runtime configuration, you must rename the file to end in `.config`.

You can find a more detailed description of patch files on the Sitecore Developer Network site. Note that this description relates primarily to Sitecore version 6.

Previewing the result of patching

To see the configuration file that is used at runtime, you load the URL http://<hostName>/sitecore/admin/showconfig.aspx.

You can also use the Config Builder in the Sitecore Instance Manager to simulate the merging process. The Config Builder merges all patch files with the `Sitecore.config` file and enables you to save the resulting file anywhere you like. You can then check whether your configuration changes have been patched in as expected.

You can find the Config Builder under Bundled Tools on the Home tab in the Sitecore Instance Manager. You must select a Sitecore site before you can select the Config Builder.

Note

The resulting configuration file shows where patched elements come from. For example, an element that was taken from the `Sitecore.Analytics.config` file includes the `patch:source="Sitecore.Analytics.config"` attribute.

Send feedback about the documentation to docsite@sitecore.net.

Contact tracking

A contact is an entry in the Experience Database (xDB) that represents an individual who is currently interacting or who may potentially interact with your organization. It contains all the information you collect about an individual as they interact with your content. You can track contacts across all devices and channels that they use.

You can track the customer experience of a contact whether they are previously identified or unidentified in the system. You can consolidate all this information as a single contact, and then personalize the customer experience further based on the behavior profile of the contact and their interaction history.

Devices

A cookie is used to identify the device that a contact uses to interact with a website. In this way, you can identify the device if the contact returns using the same device. A session cookie identifies requests that belong to the same interaction.

Every device and interaction is then associated with a contact in the xDB. When there is no contact to link to a device then a new, unidentified, contact record is created so that each device used has a contact record associated with it. When the contact has been identified, the unidentified contact is merged with the identified contact.

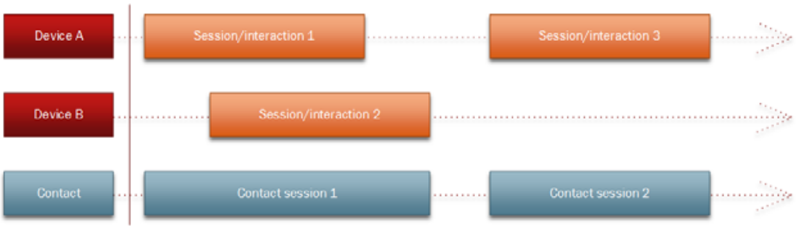
Sessions and Interactions

A session consists of a sequence of page requests, or events triggered on a website using one or more devices that take place over a specific period. These activities are recorded as an interaction in the xDB.

There are two types of sessions:

- *Device sessions* – correspond to a contact making an interaction using a device. A device session begins with the first page request but also captures other requests made during the same time period. In some browsers, closing the browser window ends the session and further actions will start a new one. In other browsers, closing the browser window has no effect, and it is possible to continue the device session until it times out (20 minutes by default). When a device session ends, all contact and the interaction data is submitted to the xDB.
- *Contact sessions* – begins together with the device session. When a contact uses multiple devices at the same time the contact session will continue as a series of overlapping device sessions. Contact information collected in one device session is immediately made available in all other device sessions. The contact session ends when the last of the connected device sessions has ended and all the data associated with the contact is saved to the xDB.

During each session, the xDB maintains contact and interaction information and makes this data available to Sitecore modules to enable personalization and website customizations.



Contact information

During interactions the following contact information is maintained and made available for personalization:

- Personal information – names, addresses, phone numbers, or email addresses from a CRM system or directly from forms on the website.
- The most recent campaigns and goals triggered, venues visited, and outcomes achieved are stored in the Key Behavior Cache.
- Contact Behavior Profile – a snapshot of contact behavior based on the pages they have visited.
- Engagement automation states – the progress a contact has made in engagement plans. For example, as a result of EXM emails or campaigns and goals triggered.
- Contact identifier - this identifier is used to identify the contact across multiple devices.
- Custom contact facets and extensions that store extra information in the xDB.
- Attachments that store extra information in the contact session without storing it in the xDB.

When a contact leaves the website, all interaction and contact information is saved to the xDB.

Identifying and merging contacts

All new contacts that interact with a website start out as unidentified contacts because the only information that has been collected at that point relates to the browser they are using. This is not enough to positively identify a contact across multiple devices.

Certain actions help to identify a contact, such as:

- Clicking a link in an email – this action connects the contact record to the owner of the email address.
- Submitting a security login form – this action connects the contact record to a password-protected user name. It makes it possible to repeatedly identify the contact regardless of the device they use to make the interaction.

When xDB identifies a contact, it merges the unidentified contact with the identified contact and the other contact record merely becomes a reference to the first one.

Merging contact data includes:

- Information from previous interactions.
- Personal data (stored in contact facets).
- Details of contact participation in engagement plans.

When a contact has been identified on multiple devices, those devices and their interactions are added to the same contact record.

Robot detection

The aim of robot detection is to filter interactions from automated browsers to avoid polluting the xDB with unwanted data. Robot detection classifies a contact based on IP address, Geo IP information, and user agent.

Robot detection relies on two heuristics to identify robots:

- A browser that does not carry cookies between requests.
- A browser that does not support JavaScript or does not raise any mouse or touch events.

Contacts

A contact represents an individual who interacts with or may potentially interact with your organization. The contact, or [contact entity](), contains all the information that you collect about an individual from their interactions across channels, devices and websites.

Tracking contacts

The Sitecore Experience Platform enables you to [track all contacts]() who interact with you, whether they are new or already known to your organization. All trackable information about an individual contact is stored in the Experience Database (xDB) collection database. Over time, this enables you to build up a detailed picture of all the experiences of every customer, enabling more accurate profiling and providing more relevant, personalized experiences.

The xDB stores information about the contact such as:

- An individual's name
- E-mail addresses
- Phone numbers
- Profile picture
- Behavior profile
- Interaction history
- Outcomes
- Engagement automation states

The xDB is fully scalable so you can track every contact without losing any important data. There are several different ways that you can [customize and extend the contact]() entity in the xDB, enabling you to collect contact data in the way that best suits the needs of your organization.

Unidentified and identified contacts

When a contact first interacts with your website they start out as an unidentified contact in the system until they provide enough information to be identified. Once a contact has been identified, the unidentified contact is replaced with the identified contact record.

You can also use the [Interaction Registry]() to import interactions recorded in other systems, such as from mobile applications, third party websites, or from a CRM system.

Viewing contacts in the Experience Profile

The Experience Profile presents key information about contacts in a single application designed for sales and marketing employees. You use this to view information about contacts, including goals converted, campaigns interacted with, events triggered, and profiles matched. The Experience Profile uses the segmentation index to retrieve a list of latest contacts from the xDB.

Example of contact and interaction data displayed in the Experience Profile.

Andrew Baker Overview

Contact Behavior Profile

The [Contact Behavior Profile]() stores information about a contact's past profiling activity such as profile pattern matches. This information is stored on the contact in the xDB, and is available each time the contact visits your website. Contact Behavior Profile information can be used to provide personalized content based on the most relevant profile pattern matches.

By default, if you use the interaction registry to import other types of interactions, then this information is used to update the contact behavior profile in the same way as for website interactions.

Key Behavior Cache

The [Key Behavior Cache]() is a cache containing a contact's historical and recent activities across all channels. These activities include, for example, campaigns visited, goals and events that have been triggered, and outcomes achieved. The Key Behavior Cache makes it possible to create and implement personalization rules based on a contact's past actions rather than just their current behavior.

The Key Behavior Cache is only available on content delivery servers.

Send feedback about the documentation to [docsite@sitecore.net]().

Create a custom contact facet

In the Sitecore Experience Database (xDB), if you want to extend the contact with your own custom, organization-specific data, then you can create custom contact facets.

A facet is a group of hierarchically organized, related attributes that describe a certain aspect of a contact. Facets are the smallest unit of attributes that can be added or removed from a contact during configuration.

Custom contact facets can have the following characteristics:

- Attributes – attributes are an element's subitems. For example, the *Address* element could contain attributes such as *country*, *state*, *town* or *email address*. Attributes can be strings, GUIDs, integers, or floating point numbers.
- Elements – an element is a data structure that contains one or more members or attributes, such as a facet.
- Dictionaries – A dictionary contains named elements. Each element in a dictionary is identified by a unique key.

  Note

  The `IFacet` interface is a marker interface that enables you to identify the root of the element tree

To create a custom facet you need to:

- Create a contract
- Implement the contract
- Configure the system to use the new facet
- Update the model configuration

Create a contract

To create a contract:

1. Create a contract to define the structure of elements or facets. The contract is the definition of what attributes you want the facet to be able to store and return. To define a contract for an element, implement an interface that extends the `IElement` interface. For facets, extend the `IFacet` interface.

   The following example defines a contract for the `Places` facet.

```
public interface ICoordinate : IElement
{
    float Longitude { get; set; }
    float Latitude { get; set; }
}

public interface IPlace : IElement
{
    string Description { get; set; }
    ICoordinate Coordinate { get; }
}

public interface IPlaces : IFacet
{
    IElementDictionary<IPlace> Places { get; }
}
```

2. To add a simple attribute such as an integer, a date-time, a GUID, a string, or a floating-point number, add a property with both a getter and a setter that returns the type you need.

   For example:

```
float Longitude { get; set; }
```

3. To add a child element, define a property with a getter only that returns the contract of the element in question.

   For example:

```
ICoordinate Coordinate { get; }
```

4. To add a dictionary or collection of child elements, define a property with a getter only. For example:

```
IElementDictionary<ICoordinate> Coordinates { get; }
IElementCollection<ICoordinate> Coordinates { get; }
```

   Dictionaries and collections of simple attributes are not supported. For example, if you need a dictionary of integer values, you can define an element with a single integer attribute and use this element in the collection or dictionary as the examples in this section show.

Implement the contract

To implement the contract:

- Register the attributes, collections, and dictionaries with the base class in the constructor using the following helper methods:

```
this.EnsureAttribute<TValue>( string name );
this.EnsureElement<TElement>( string name );
this.EnsureDictionary<TElement>( string name );
this.EnsureCollection<TElement>( string name );
```

These methods ensure that the member is registered with the model. Implement the properties of the contract as shown below:

```
[Serializable]
internal class Coordinate : Element, ICoordinate
{
  private const string LONGITUDE = "Longitude";
  private const string LATITUDE = "Latitude";

  public float Longitude
  {
      get
      {
          return this.GetAttribute<float>( LONGITUDE );
      }
      set
      {
          this.SetAttribute( LONGITUDE, value );
      }
  }

  public float Latitude
  {
      get
      {
          return this.GetAttribute<float>( LATITUDE );
      }
      set
      {
          this.SetAttribute( LATITUDE, value );
      }
  }

  public Coordinate()
  {
    this.EnsureAttribute<float>(LONGITUDE);
    this.EnsureAttribute<float>(LATITUDE);
  }
}
```

Note

Make sure the class is marked with the `[Serializable]` attribute. This attribute is required to ensure that instances of this class can be stored in shared session and in the submit queue.

Do not add any additional code to this implementation. The implementation classes for this example have not yet been implemented.

Configure the system to use the new facet

Once you have defined and implemented the facet structure, you must configure Sitecore to use the new facet.

- Open the `Sitecore.Analytics.Model.config` configuration file and register all elements. Elements should only be registered once even if they are used in multiple facets.

  Note

  Facets should be registered in the same way as all other elements.

```
<model>
  <elements>
    <element interface="type" implementation="type" />
  </element>
</model>
```

  In the `Sitecore.Analytics.Model.config` file, the *interface* attribute is the type defined in the contract the data model, such as `ICoordinate`, `IPlace`, or `IPlaces`. The *implementation* type is the actual implementation of the contract that is instantiated by Sitecore to hold the data in memory.

Update the model configuration

To update the model that you want to extend:

- Locate the entity type to be extended and register the new facet:

```
<model>
  <entities>
    <entity>
      <facets>
        <facet name="name" contract="type" />
      </facets>
    </entity>
  </entities>
</model>
```

  In this example, the node `<entity>` is replaced with the entity type that you are extending, for example, `<contact>`. The *name* attribute is the name under which the facet is accessible on the contact model. This is the string value passed to the `GetFacet<TFacet>( string name )` method.

```
IPlaces places = contact.GetFacet<IPlaces>( "Visited Places" );
```

Send feedback about the documentation to docsite@sitecore.net.

Architecture options

When you install the Sitecore Experience Platform with the Sitecore Experience Database for the first time, there are three main architecture options that you can choose from:

- On-premise server solutions
- Hybrid server solutions
- Cloud server solutions

Whether you choose to run Sitecore as a cloud, on-premise, or hybrid solution depends on the type and size of your organization. For example, a small organization may only need a few servers and therefore a public cloud solution could be suitable, whereas a large enterprise organization may need its own data center.

On-premise server solutions

In an on-premise solution, you install all your components, such as Sitecore application servers and MongoDB, on your own physical servers.

You can install Sitecore as an on-premise solution in one of the following ways:

- As a single-server, standalone environment (not scalable). All components are on a single server computer.
- As a multi-server, scalable environment (fully scalable). The multi-server environment consists of servers for each Sitecore component, for example, content delivery, content management, collection database, session state, reporting service, reporting database, and processing servers. You also have the option of expanding this environment to handle big data by implementing vertical and horizontal scaling.

Note

Only install a standalone environment for development or testing purposes. For a production environment, you should not install all xDB components together on a single server. A standalone environment is not scalable and cannot offer the same performance as a multi-server environment.

Advantages and disadvantages of on-premise server solutions:

Advantages	Disadvantages
More control Unlimited configuration options Potentially unlimited hard disk space depending on budget Can choose whether to use solid state drives (SSD) or hard disk drives (HDD)	Complicated hardware setup Costly to create and maintain Time-consuming to administer Costly and time-consuming to expand/upgrade and scale storage Requires multiple data centers for geographically distributed failure zones - easier and cheaper than public cloud

Note

Public cloud providers can also offer unlimited disk space. The limitation in both on-premise and cloud solutions is your budget. It is much easier and faster to scale storage in the public cloud than it is within your own premises.

Hybrid server solutions

In a hybrid solution, you install some components on-premise while others components are in the cloud. For example, a hybrid server solution could consist of the following:

- A multi-server, scalable Sitecore environment installed on site (see the previous section for more detail about on-premise solutions). This can include content delivery, content management, processing, and reporting servers set up at an organization's data center.

- MongoDB collection database server in the cloud (see the next section for details).

Advantages and disadvantages of hybrid solutions:

Advantages	Disadvantages
MongoDB in the cloud – less need for specialist MongoDB knowledge and expertise – no new staff or training required Ease of maintenance and upgrade.	Less control over scaling the collection database to handle large numbers of contacts and visits Possible limitations when using MongoDB with some private cloud solutions - limited amount of data storage

Cloud server solutions

In a full cloud solution, all Sitecore Experience Platform components are in the cloud.

For example, a fully scalable cloud environment – this includes all Sitecore xDB components, such as content delivery, content management, processing, reporting servers, session state, and MongoDB (collection database).

Advantages and disadvantages of cloud solutions:

Advantages	Disadvantages
Simplifies installation and configuration No specialist knowledge needed Globally distributed data centers – ensuring high availability Automatic backup No need to manage scalability Security - spread data across multiple locations	Cost Less control – over hardware and scalability strategy Not always suitable for large organizations that have their own data centers - therefore, a hybrid solution may be a good option

Send feedback about the documentation to docsite@sitecore.net.

Architecture overview

The Sitecore Experience Database (xDB) collects all your customer interactions from all channel sources in a real-time, big data repository. It connects interaction data to create a comprehensive, unified view of each individual customer, and makes the data available to marketers to manage the customer experience in real time.

The xDB architecture is flexible allowing you to deploy single-server, low-traffic solutions as well as highly scalable, high-traffic, multi-server, solutions.

Key components

The key architecture components in the xDB are:

- Content delivery server (including personalization)
- Content management server
- Content databases (SQL Server)
- Session state server
- Collection database (MongoDB)
- Processing server
- Reporting database
- Reporting service

Content delivery server and personalization

You can implement content delivery and personalization using a Sitecore application server that serves incoming HTTP requests for website content from the Internet.

Content management server

Content management in this document refers to the core functionality of Sitecore such as authoring and editing content, managing content, and configuring your website solution.

CMS databases (SQL Server)

Sitecore CMS uses the following three SQL Server databases:

- *Core* – contains all settings, like a large configuration file for the Sitecore user interface. You can use this if you are customizing Sitecore, for example, by adding new applications to the Sitecore desktop.
- *Master* – contains all versions of all content. This is where business users author and edit content.
- *Web* – contains the latest version of published content that has reached a final workflow state.

Session state server

The session state component is a session state store used by the content delivery and personalization process. This is a standard ASP.NET session state store provider that includes support for the `Session_End` event. In Sitecore xDB, the default session state provider is `inProc` ASP.NET but it also comes with two additional providers, one for MongoDB, and another for SQL Server.

Collection database (MongoDB)

The collection database (MongoDB) is the primary storage for all analytics information and the registry of contacts and engagement automation states. It uses MongoDB – a highly scalable document-based NoSQL database solution.

Processing server

The processing and aggregation component extracts information from captured, raw analytics data and transforms it into a form suitable for use in reporting applications. It also performs specific tasks on the collection database that involve mass updates.

You implement processing and aggregation on a Sitecore application server connected to both the collection and reporting databases. A processing server can run independently on a dedicated server, or on the same server together with other Sitecore components. By implementing multiple processing or aggregation servers, it is possible to achieve higher performance on high-traffic solutions.

Reporting database

The reporting database is a SQL Server database that stores aggregated data from the collection database suitable for fast querying and reporting.

Reporting service

The Reporting Service API allows you to execute queries and extract information from the collection and reporting databases. It is a part of the xDB but you can also configure the Reporting Service as an intermediate reporting server which performs processing tasks and offloads endpoint application servers.

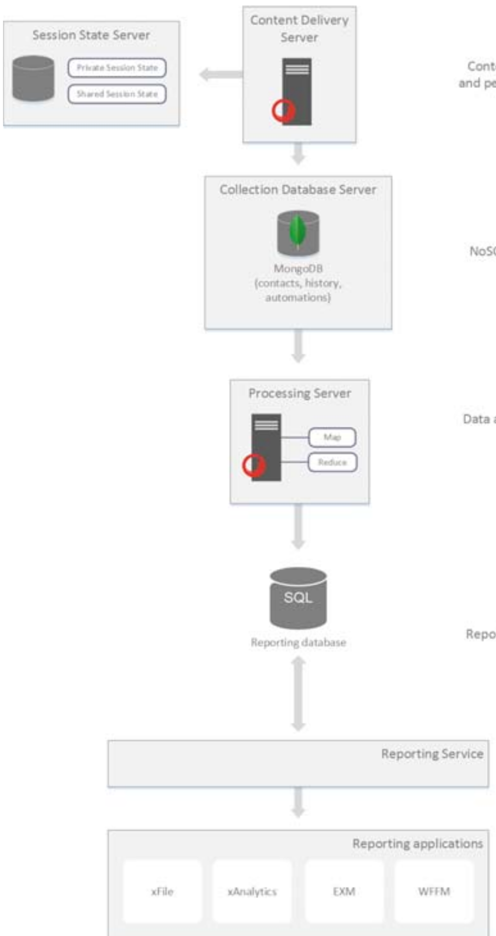
Note

The database names collection and reporting are not official Sitecore product names.

The collection database refers to the functionality and purpose of the MongoDB analytics database.

The reporting and reporting.secondary databases correspond to the SQL Server *Sitecore_Analytics* and the *Sitecore_Analytics_Secondary* databases. These are legacy names that continue to be used.

Diagram showing a simplified version of the xDB architecture



xDB data flow

When a contact interacts with your website, the xDB collects all contact and the interaction data. This data flows through the xDB in the following sequence:

1. A contact decides to interact with a website.
2. Depending on the geographic location of the contact, the xDB redirects their visit to the closest cluster (data center) through DNS configuration.

3. Load-balancing software directs the contact to an appropriate Sitecore content delivery server. Contacts stay connected to the same cluster even if they switch devices or use another browser. Contacts can only move to another cluster if all their active sessions expire and all their information is then saved to the collection database.
4. During the session, details of the contact, the interaction, and the device are stored in either a private or shared session state. On session end, Sitecore flushes this data to the collection database and schedules it for processing.
5. The processing layer reduces and groups interaction (online visit) data and stores it in the reporting database for use by the reporting layer. This is known as aggregation. Data is also aggregated by the processing layer for use by the segmentation index.
6. The Reporting Service queries the reporting database to fetch reporting data for use in applications such as Experience Analytics (xAnalytics).

Send feedback about the documentation to docsite@sitecore.net.

Collection database overview

In the Sitecore Experience Platform, the collection database (MongoDB) is a highly scalable database capable of collecting and storing vast amounts of customer experience data. In the Sitecore Experience Database (xDB), the collection database is the central repository for storing contact, interaction, history, and automation data.

The collection database can help you to increase the availability, scalability, and performance of your Sitecore implementation to handle billions of visits.

In the xDB, MongoDB is the default choice for the collection database because it is a highly scalable database solution. MongoDB is an open-source, NoSQL, document-oriented database, designed for ease of development and scaling.

The main benefits of using MongoDB with the xDB are to:

- Handle billions of visits or interactions per year.
- Provide scalability options to increase storage capacity.
- Maintain high performance.

The collection database (MongoDB) collects and processes analytics data from all your websites. This data can include interactions, contacts, devices, location, automation data, and events that contacts trigger, such as goal conversions or campaigns.

MongoDB and NoSQL

MongoDB and other document-oriented database systems require you to think about databases in a slightly different way. Previously in Sitecore, all visitor analytics data was collected and stored in Microsoft SQL Server. This approach works well but, over time, extracting relevant data for reporting purposes becomes more and more time consuming and expensive, particularly when there are billions of visitors.

In SQL Server, it is possible to scale vertically (increasing the size of the server) but it is not so easy to scale horizontally (increasing the number of servers). So to achieve the required improvements in scalability, performance, and availability, a solution such as MongoDB is the appropriate choice.

Before you implement Sitecore with MongoDB as your collection database, see MongoDB considerations for guidance on MongoDB architecture and hardware configuration.

Send feedback about the documentation to docsite@sitecore.net.

xDB hardware guidelines

When you install the Sitecore Experience Database (xDB) as an entirely on-premise solution, you need to consider the minimum hardware requirements for each instance in your environment, depending on the roles you want to allocate to each server. This topic outlines the hardware requirements and describes an example hardware configuration.

Note

The example hardware configuration described in this topic aims to achieve the best performance possible but has not been tested for every possible scenario. You need to perform tests to find out what suits your system best.

Content management and content delivery servers

The hardware requirements for content management and content delivery servers remain the same as in earlier versions of Sitecore and are described in the manual Installing the Sitecore Experience Platform on the Sitecore download website.

Session state server

The hardware requirements for session-state servers depend on which session-state service you choose to use. If you choose to install a session-state server with the xDB, you can configure this to use a Microsoft SQL Server or MongoDB database. MongoDB installation instructions with software and hardware requirements are described in full on the MongoDB website: MongoDB installation guides

Note

You should use a fast network, solid-state drive (SSD) on your session-state server and plenty of RAM.

Collection database server

Using MongoDB as your collection database, you should install plenty of RAM and use SSD drives. Sharding can also improve performance significantly. Read the documentation on the MongoDB website to learn about the MongoDB architecture, replication, sharding, and configuration options.

Processing and aggregation server

Processing and aggregation servers can use RAM for caching and for establishing a fast connection to the databases. The more cores the CPU has, the more agents you can configure to run in parallel.

Reporting server

The hardware requirements for the reporting database server have not changed significantly since Sitecore 7.5 (see the *Hardware configuration example* table).

Better optimization of the reporting database means that it now uses memory more efficiently. However, you should install fast SSD hard drives and a minimum of 14 GB of RAM to achieve the best performance.

Read the Microsoft SQL Server documentation on MSDN to learn more about configuring Microsoft SQL Server.

Hardware configuration example

The following hardware example was implemented on a simple website when setting up the xDB for the first time.

Important

There are many alternative ways of installing and configuring Sitecore xDB. The hardware details in this section are just a minimum example. They are not a set of recommendations to follow.

You can use the information in this table to get an idea of what you might need when setting up the xDB for the first time.

Server type	CPU	RAM	Disk
Content delivery	4 x CPU E5 2650 v2 processors. The more cores, the better.	16 GB (including the operating system)	HDD 40 GB per instance
Processing	4 x CPU E5 2650 v2 processors. The more cores, the better.	16 GB (including the operating system)	HDD 40 GB per instance
Content management	4 x CPU E5 2650 v2 processors. The more cores, the better.	16 GB (including the operating system)	HDD 40 GB per instance
Session state (Microsoft SQL Server or MongoDB)	4 x CPU E5 2650 v2 processors	4 GB The more, the better	SSD shares the hard disk with the collection database – ideally the hard-disk capacity should not be more than the RAM, which is 4GB in this example.
Collection database (MongoDB) 2 servers	4 x CPU E5 2650 v2 processors	16 GB The more the better	SSD 100 GB The more disk space you have, the more data you can store.
Reporting database server (SQL Server)	4 x CPU E5 2650 v2 processors	16 GB The more, the better	SSD 100 GB The more, the better
Search and indexing	4 x CPU E5 2650 v2 processors. The more cores, the better.	12 GB (including the operating system)	HDD 100GB - SSD may offer better indexing write throughput. - Use SOLR Cloud (clustering) to spread disk writes across multiple disks to scale out.

Note

If you do not use Solr for search and indexing, then using Lucene may require between 100 and 200 GB of hard disk space. Depending on the type of server you are configuring you should adjust disk space and memory accordingly.

HDD = hard disk drive

SSD = solid state drive

RAM

On each xDB server instance you should install plenty of RAM, particularly on collection, processing, reporting, and session-state servers as outlined in the *Hardware configuration example* table.

Operating system

Include the Windows operating system in the RAM usage because it is difficult to predict accurately how much RAM the Windows operating system will consume.

Session

The more RAM you have, the more sessions you can track simultaneously without using extra disk space. Actual performance, however, depends on the specific configuration you have. You should avoid disk I/O on reads because this can have a negative effect on performance.

Session database

The session database should ideally fit 100% into RAM, otherwise it gets too slow.

Collection database (MongoDB)

RAM serves as disk cache. You need at least enough RAM to store a working set size (the MongoDB website has instructions on how to estimate this).

MongoDB FAQs and diagnostics

Hard disk

In general, the more disk space you have, the more data you can store. Refer to the *Hardware configuration example* table in this topic for more specific guidelines.

Send feedback about the documentation to docsite@sitecore.net.

MongoDB considerations

MongoDB is a highly scalable document-based NoSQL database solution that the Sitecore Experience Database (xDB) uses for the collection database. Before you install MongoDB, you should consider the following:

- Decide whether you want a public cloud-based solution, such as Windows Azure, or one within your own data center on either physical or virtual hosts. For more information, see

  MongoDB fundamentals

- Ensure that the MongoDB working set fits in memory for optimal performance. This is often the most important performance factor when you are considering vertical scaling or scaling up, followed by disk and CPU. For more information about working sets and RAM, see:

  MongoDB storage

  MongoDB RAM

- Apply security practices; The default MongoDB configuration assumes a trusted networking environment and cannot be exposed to a public service such as Azure, this is described in the MongoDB Production Notes. When deploying on Azure or another public infrastructure service, always take the necessary security measures described in the MongoDB Security Checklist, or consider using a managed MongoDB service.

  Note

  Avoid deploying publicly accessible MongoDB instances without enabling authentication or encryption.

- Select the right hard disks – you can use solid-state drives (SSD) for fast reads and writes, or hard disk drives (HDD) for larger capacities at lower cost. Typically, you should use SSDs as they offer significantly higher performance. If you use HDDs, use 15k RPM SAS disks in a RAID10 configuration for a balance of performance and fault tolerance.
- Decide how many MongoDB servers you need:
  - Standalone server – for testing and development
  - Single replica set – the minimum recommended configuration for production, which should have at least 2.5 servers: two full capacity data servers for failover and one low capacity server for the arbiter. You should use three data servers for a robust and resilient deployment, especially during maintenance.

    For more information, see: MongoDB replication.

- Read the whitepaper *Performance Considerations for MongoDB*:
  - MongoDB Performance Considerations 2.4 PDF

For more information about MongoDB, see their documentation.

Note

All production deployments of MongoDB should use replication, which requires a minimum of three servers, to provide redundancy and increase data availability. You should only use single server installations for development and testing.

Send feedback about the documentation to docsite@sitecore.net.

MongoDB example architecture

This topic provides an example of the standard approach to configuring MongoDB when you install the Sitecore Experience Database (xDB) with MongoDB as your collection database. This example includes the minimum possible number of MongoDB instances in a single replica set while still ensuring that replication is in place. This setup consists of three Azure virtual servers: one primary and two secondary.

Replication

All production deployments should use replication to provide redundancy and increase data availability. Replication copies data to multiple servers, so if one server fails no data is lost. You can place servers in different geographical regions, defaulting to one and using the other as backup for disaster recovery (DR) or high availability (HA) failover.

Setting priorities

Depending on how you configure MongoDB, you can increase read capacity by setting the priority on each instance. The primary is the only member of a replica set that can receive write operations. You set priority by allocating a number to each instance, the highest number gets top priority.

In the example in this topic, the member with priority 10 will always be elected the primary if it is available. The other member with priority 9 will be secondary. If you have an arbiter this has no priority setting, as it can never become the primary (it does not store data and its only real use is for voting during elections). Similarly, a data member with priority 0 will never become primary, though it still maintains a copy of the data and can serve reads if the appropriate read preferences are used.

If the primary becomes unavailable for any reason, MongoDB performs an election to determine which member becomes the new primary. Only members that have a value higher than zero (0) can take part in elections and there must be a majority vote.

You can use priorities in many other useful ways, for example, to give priority to nodes with the most powerful hardware or to give priority to the primary data center.

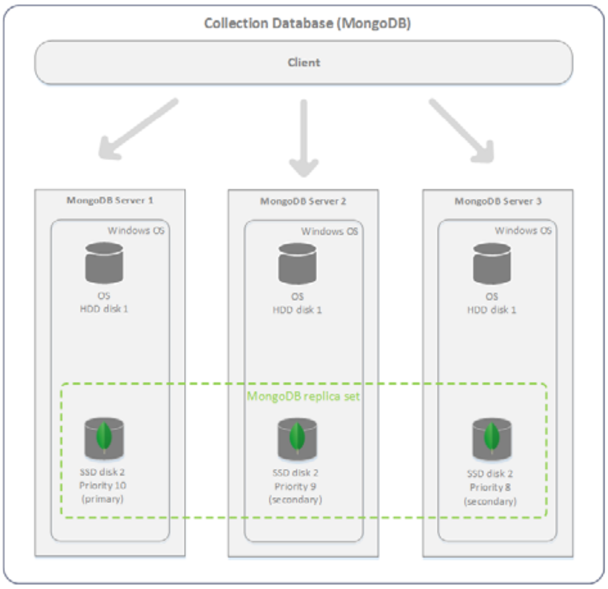
Note

You should not set priorities just to make it more predictable which node will become the primary because this can result in making operations more complicated. This information can easily be discovered using the `rs.status()`method.

Refer to the MongoDB documentation for more information on replica sets, setting priority, and elections: [MongoDB replication]

Example architecture

The Collection Database (MongoDB) diagram shows an example of a standard MongoDB replica set implementation on three servers (3 data nodes). For reliability and resilience, you should have 3 data nodes (1 primary, 2 secondary), especially if write concern w=majority is used (to avoid rollbacks) and during periods when one of nodes are unavailable (for example, during maintenance, upgrading, or defragmenting).



Sharding

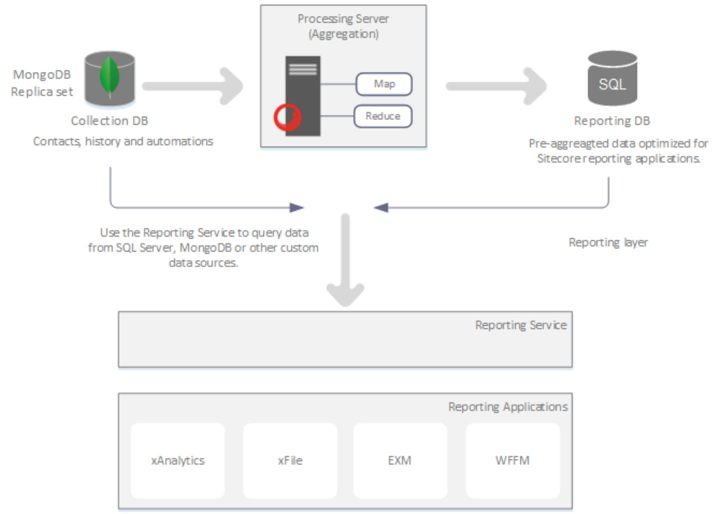
As the collection database grows, you will find that you need more storage space. You might also find that read and write speeds slow down. In MongoDB there are different ways of configuring your environment to cope with these demands. Sharding enables you to scale horizontally. This means adding more computers to store data and spread the load, which can also help to improve read capabilities, write speeds, and contributes to generally improved performance.

For more information on installation, replication, and the different configuration options available in MongoDB, see the [MongoDB website].

Send feedback about the documentation to docsite@sitecore.net.

Reporting architecture

The Sitecore Experience Database reporting architecture is a highly scalable and high-performance solution for storing data and delivering reports.



In the Sitecore Experience Database, the reporting architecture consists of:

- Reporting server – a Sitecore server that hosts the Reporting Service
- Reporting Service – uses queries to fetch report data from various data sources, such as MongoDB, SQL Server, or other custom data sources, such as a customer relationship management system (CRM)
- Reporting database – a Microsoft SQL Server database that contains fact and dimension tables of aggregated data from the collection database
- Reporting applications – for example, the Experience Profile (xFile) and Experience Analytics

In earlier versions of Sitecore, there was only a single analytics SQL Server database for all collection and reporting needs. In Sitecore xDB, for high traffic websites, SQL Server is no longer scalable or high performance enough to perform all data collection and reporting tasks.

To improve scalability and performance, the xDB uses the following database systems:

- MongoDB (NoSQL) collection database – stores all contact and interaction data
- Microsoft SQL Server database – Sitecore reporting applications read data from a SQL Server reporting database

Note

You should install the SQL Server reporting database on a dedicated server to achieve higher performance.

The xDB collection database is a scalable, document-oriented database that can handle billions of interactions. It collects and stores all visitor data, such as contacts, devices, locations, goals, engagement automation states, and other details. By default, it uses MongoDB to store all relevant information in documents that are roughly equivalent records in a relational database but it is quicker to query, so writing and reading is very fast. However, in MongoDB, the data operations required for reporting are either not supported or only partially supported.

Send feedback about the documentation to docsite@sitecore.net.

Scalability options

In the Sitecore Experience Database, to achieve better performance and to cope with greater demand and large amounts of website traffic, you have several scalability options.

You can configure Sitecore xDB environments to:

- Run as a standalone environment
- Scale vertically
- Scale horizontally

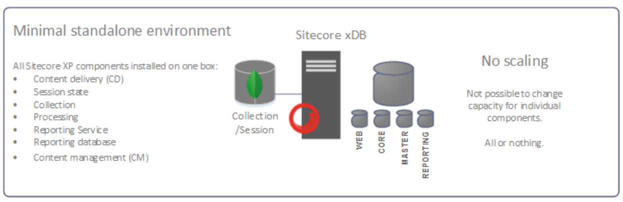
This topic outlines three possible approaches that you can implement to achieve better performance and scalability.

Standalone environment

A standalone or minimal installation is an all-in-one configuration where you install all Sitecore xDB components on the same computer.

For example, a minimal installation could include the following components:

- A Sitecore application server instance with all the components necessary for content management, content delivery, processing, reporting, and session tracking
- A collection database (MongoDB) – to record the entire customer experience
- A reporting database (SQL Server) – to provide reporting data for the Experience Profile and Experience Analytics reports
- Content databases (Master, Core, and Web)



You should not use a standalone installation as a production environment as this setup is most suitable for development or testing, and is the least scalable approach.

Vertical scaling

Definition: "*To scale vertically (or scale up) means to add resources to a single node in a system, typically involving the addition of CPUs or memory to a single computer.*" (Wikipedia)

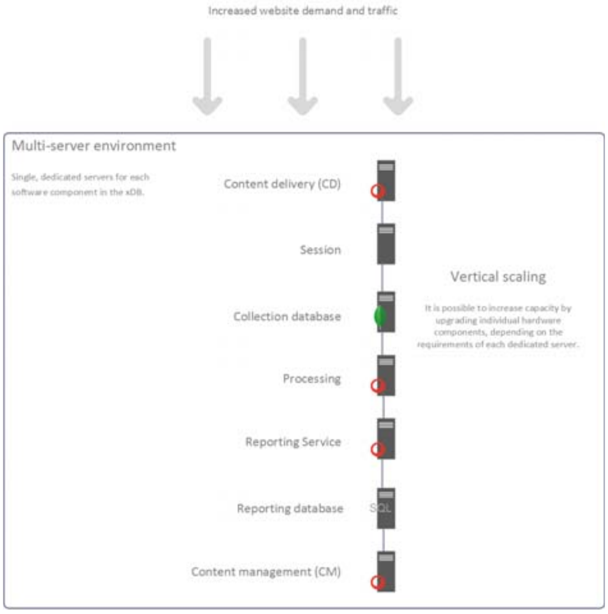
In Sitecore xDB, you could start the process of vertical scaling by installing single instances of each component on separate servers, therefore moving from a single server to a multi-server environment.

For example, you could start by having the following three servers:

- Content delivery
- Collection
- Reporting

An advantage of vertical scaling is that you can upgrade specific components (depending on role) without upgrading the entire environment. For example, a content delivery server might need faster CPUs, while the collection database needs more disk space for storage, and plenty of RAM to maintain system performance. Therefore, if you scale vertically, you can dedicate specific servers to specific roles, and upgrade hardware accordingly.

Sitecore xDB vertical scaling example



Vertical scaling is suitable for the needs of most organizations, particularly small to medium-sized organizations where the data requirements are not too high.

For example, you could move from a single server to a multi-server environment by:

- Deploying collection, reporting, and content databases on separate servers.
- Deploying application components, such as content delivery, content management, and processing on separate servers.

Horizontal scaling

Definition: "*To scale horizontally (or scale out) means to add more nodes to a system, such as adding a new computer to a distributed software application.*" (Wikipedia)

In Sitecore, horizontal scaling means you deploy multiple servers for specific components, such as content delivery, processing, content management, or collection, to increase the capacity of your solution.

For example, as demand increases, you can deploy multiple clusters of content delivery servers to deliver web content to contacts as quickly and efficiently as possible. On multiple content delivery clusters, you can use load-balancing software to optimize performance across clusters and to maintain high availability. You could also implement multiple content delivery clusters to provide multiple geographically-distributed collection points.

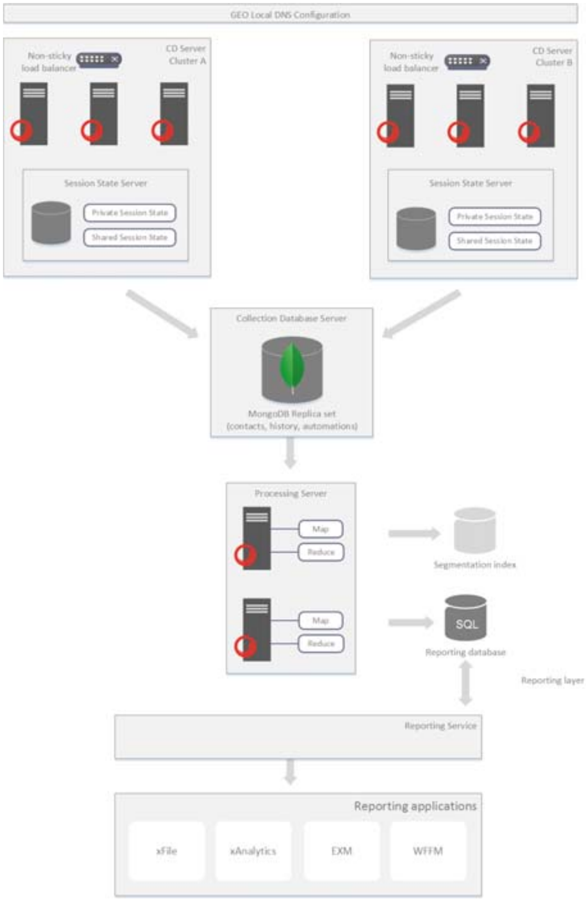
Implementing a fully scalable xDB architecture brings many benefits, such as improved performance and high availability. This approach is particularly suitable for the largest, enterprise organizations that need to handle a lot of website traffic and want reliability and high availability. These types of organizations often have large data storage requirements so may need to consider horizontal scaling as part of their future strategy for expansion.

For example, to scale horizontally you can deploy:

- Multiple content delivery servers.
- Multiple clusters of content delivery servers, either to increase capacity beyond a single cluster, or to create geographically-distributed content delivery and traffic collection points.
- A dedicated session state server for handling session data. This is a requirement if you have a cluster with more than one node.
- A collection database (MongoDB) for storing contact and interaction data – this can be a single MongoDB replica set (which requires a minimum of three nodes, for example, two data nodes and an arbiter) or it can be a sharded cluster with multiple replica sets.
- One or more processing servers. The number of processing servers you have depends on how much website traffic you need to handle.
- Multiple content management servers – depending on your content editing needs.
- A reporting server that runs the Reporting Service hosted on a dedicated Sitecore server instance.
- A reporting database hosted on a dedicated Microsoft SQL Server instance (this could also be a failover cluster).

Sitecore xDB - fully scalable architecture (horizontal scaling)

This diagram shows how you could start to implement horizontal scaling (scaling out) by configuring multiple content delivery clusters. It shows how you can add more nodes to your xDB architecture as demand on your website increases while at the same time maintaining high performance.



Benefits of a fully scalable xDB architecture

The xDB is particularly suitable for medium and large-scale enterprises. The main benefits of deploying a fully scalable Sitecore Experience Database architecture, include:

- *High flexibility* – from a single server to highly scalable solutions with many different configuration options. As demand on your website increases, you can expand your solution by scaling up and scaling out.
- *High performance* – for example, you can optimize the performance of multiple content delivery servers using load balancing software and have the possibility to scale almost any component vertically and horizontally.
- *Full support for content delivery web clusters* - you can configure content delivery web clusters and session state right out of the box, without the need for any extra custom components. You can distribute the load on your servers using load-balancing software with failover, and can upgrade your solution while it is still live.
- *Highly scalable data storage* – a MongoDB database at the core of data collection provides storage that incrementally scales to terabytes or even petabytes of data. You can increase capacity by horizontal scaling, which in MongoDB you can implement by sharding. You can add more servers over time when you need to expand your solution. It is possible to implement both replication and sharding across multiple geographical locations. All customer interaction data (online visits) is stored in a MongoDB NoSQL database (nothing is deleted) and is made available to Sitecore reporting applications. Also, you can scale out the processing server role to perform all activities necessary for processing and aggregation.
- *High availability* – all application components can be deployed as multiple servers, and database components support high availability and data persistence. In MongoDB, replication enables higher availability (HA) and better fault tolerance.
- *Cloud-ready* – you can deploy application components as preconfigured servers allowing for quick scaling up and down on demand, depending on traffic patterns.

Send feedback about the documentation to docsite@sitecore.net.

xDB software recommendations

For each Sitecore Experience Database server instance that you install, you need the following software:

- Microsoft .NET Framework 4.5.
- Windows Server 2012 R2 – recommended for most Sitecore instances, especially processing/aggregation servers.
- MongoDB for the collection database, session storage, and tracking databases – when you install MongoDB, ensure that it runs as a service.
- Microsoft SQL Server 2008 R2 SP1 or higher for the reporting database.

Note

You should also be familiar with the software requirements for a standard Sitecore CMS installation.

Important

Sitecore neither supports nor permits multiple IIS web site definitions pointing to the same Sitecore web root.

You can find information on Sitecore XP compatibility with different versions of MongoDB in the Sitecore Knowledgebase.

Verify a connection to the collection database

When you install the Sitecore Experience Database for the first time, you do not have a pre-defined website. To test that you have a connection to MongoDB, you need to generate some local visits to your blank website. Any visits you make should appear as new interactions in the collection database.

To test your Sitecore CMS connection strings, first open your website in a web browser and check that you can log in to the Sitecore Desktop.

To verify that you have successfully made a connection to your MongoDB collection database you have to:

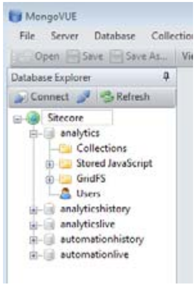
- Open MongoDB and make a local visit to your website
- Change the session end default timeout setting
- Check MongoDB for interaction documents
- Verify that fact and dimension tables contain data

Open MongoDB and make a local visit to your website

To check that interactions appear correctly in the collection database:

1. Start your MongoDB database, open your MongoDB client, for example MongoVUE, and connect to the appropriate database.
2. In MongoVUE, you can see that before any contacts visit your website, only some collections are visible.

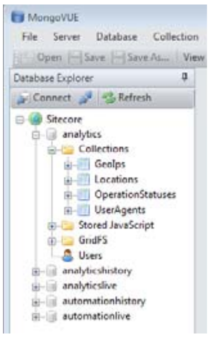
   For example, in the following screenshot, the Analytics database appears but it does not contain any collections.

   The Contacts and Interactions collections are missing until your website has visits.

3. Make a visit to your website test page and then refresh MongoDB.

   Expand the analytics database and then expand the Collections node.

In the example, you can see four new collections relating to browser and system information but there are no interaction or contact collections.

Change the session end default timeout setting

After making a test visit, you need to wait 20 minutes before the session times out and MongoDB updates with the latest changes. For testing purposes, you may want to change this default timeout setting to make data appear more quickly.

To change the default timeout setting:

1. In the *Website* folder, open the `web.config` file.
2. Navigate to the `<sessionState` node.
3. Change the session end default timeout setting from *20* to *1* minute.

```
<sessionState mode="InProc" cookieless="false" timeout="1">.
```

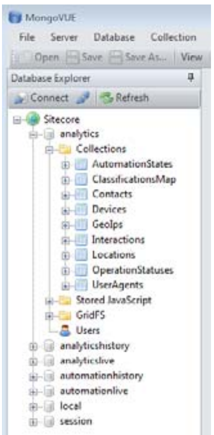
Note

It is important that you only change the default timeout setting for testing purposes. For optimum performance restore the default session end timeout setting as soon as possible.

Check MongoDB for interaction documents

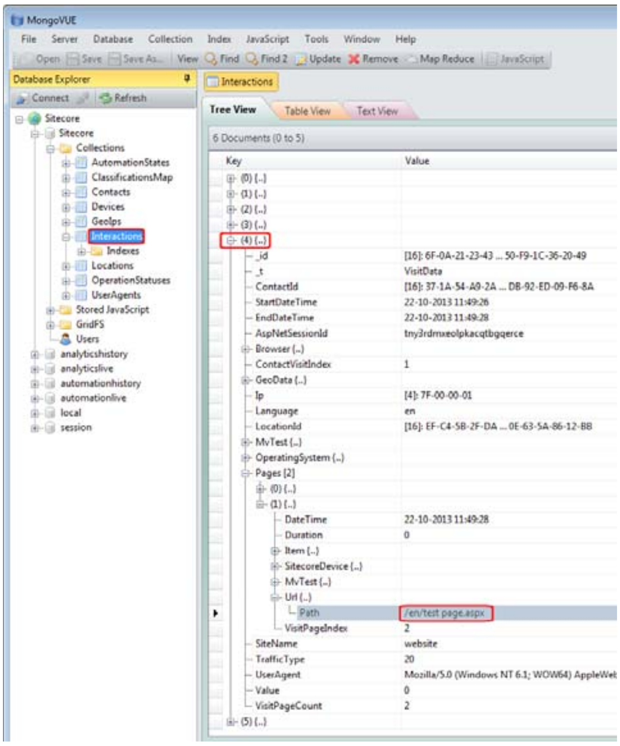
After changing the default session end timeout setting, make a visit to your website and then refresh MongoDB. Interaction data should now appear in your MongoDB documents after one minute.

1. Refresh the MongoDB database again. After a short time, you should see the Contacts, Devices, and Interactions collections added to MongoDB.



2. Right click Interactions and then click View to see all the interactions on your website.

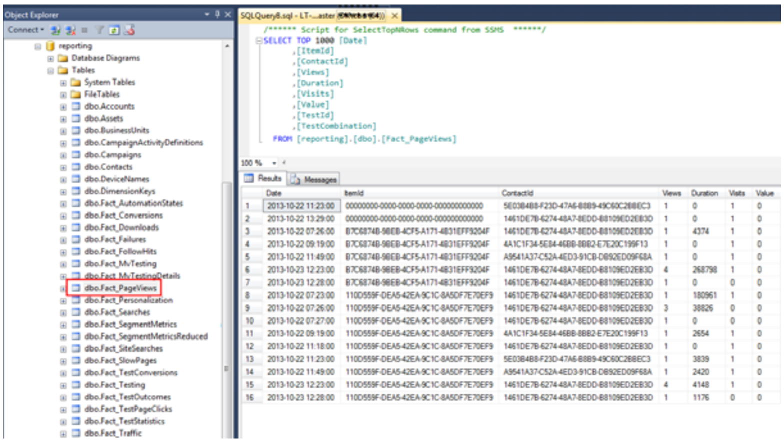
   In the following example, five interactions are stored in the collection. Expand one of these interactions to view data relating to the pages visited. The visit in the following screenshot consists of two page views.



Verify that fact and dimension tables contain data

Before you can verify that fact and dimension tables contain pre-aggregated data, ensure that you have data stored in the collection database and have configured your processing server correctly.

- Query one of the SQL Server fact tables, such as `Fact_PageViews`, and view all rows to check that it contains data. If the table contains analytics data, the processing layer is working correctly.



Note

If you do not see any data in the `Fact_ tables`, check the `ConnectionStrings.config` on the processing server to see if the analytics connection string to MongoDB is configured correctly and that you have data in the collection database.

Send feedback about the documentation to docsite@sitecore.net.

Batch aggregation configuration

Batch aggregation enables you to group interactions together into batches to improve the performance of your aggregation processing framework. In the Sitecore Experience Database (xDB), batch aggregation is part of the standard installation, but you can also change the default settings depending on your specific hardware and solution architecture. You can test the performance of your batch aggregation configuration using the performance counters in this topic.

You should change the default batch aggregation configuration if you want to:

- Improve the performance of batch aggregation to suit your solution.
- Enable batch aggregation to use your own custom aggregations.

To change the default batch aggregation configuration you need to edit the `Sitecore.Analytics.Processing.Aggregation.Services.config` file.

Note

Do not make changes directly to the configuration file, but instead create your own custom configuration patch file that performs the required changes during run time.

Batch aggregation configuration settings

The following settings enable you to adjust the configuration of batch aggregation to improve the performance of your Sitecore solution during aggregation processing.

Configuration node	Description and example
`MaximumBatchSize`	The `MaximumBatchSize` setting enables you to control the maximum number of interactions to include in a single batch. Adjust this setting according to the performance of your individual Sitecore solution. Consider the following factors when you set the maximum batch size: - The number of SQL Server disks you have. - Processing power (type and number of processors). - Network latency. - The specific configuration of your aggregation/processing server. The default setting is 64 interactions: `<MaximumBatchSize>64</MaximumBatchSize>`
`MultiplexingTimeout`	Multiplexing collapses multiple aggregation threads into a single transaction just before commit to make processing faster and to reduce the load on SQL Server. The multiplexing time out setting enables you to specify the maximum time the multiplexer can wait for parallel executing aggregation agents before storing the data set in the reporting database. Setting the value to 1 second (default value) tells the system to save the combined results produced by aggregation agents each second. The default setting is 1 second: `<MultiplexingTimeout>0.00:00:01</MultiplexingTimeout>`

`Aggregator`	Replace the standard implementation of the batch aggregator with your own custom implementation.
	`<Aggregator>`
	`  <MultiplexingTimeout>0.00:00:01</MultiplexingTimeout>`
	`</Aggregator>`

Table-valued parameters (TVP)

Sitecore uses table-valued parameters (TVP) to send aggregated information to SQL server.

For information on how to use table-valued parameters read the Microsoft article Use Table-Valued Parameters (Database Engine).

Using TVP provides a faster way of communicating with SQL Server. It enables you to pass a table as a parameter to a function or stored procedure after the data from the collection database has been grouped together. The use of TVP is optional and is implemented separately for each fact and dimension table.

The SQL table mappings in the Sitecore configuration file have the following properties to support TVP:

- `IsMultiRow` – shows whether the stored procedure accepts TVP.
- `TableType` – the name of the SQL Server user-defined type that is accepted by the stored procedure. This type must correspond to the structure of the table. It must have the same columns in the same order and use the same types. For more information on user-defined types, read the Microsoft article Working with User-Defined Types in SQL Server.

SQL table mappings are located at the following path: `/sitecore/aggregation/routines/ExecRoutineStatementBuilder/mappings` and defined in the `Sitecore.Analytics.Processing.Aggregation.config` file.

It is recommended that you create custom aggregations that support for TVP. See the example for a description of how to configure Sitecore to do this.

Configuration patch file examples

You can customize the following examples to change your default batch aggregation configuration.

Example 1: Changing the default configuration settings

This is an example of a configuration patch file that you can use to change the default batch aggregation configuration file settings. It shows how to change the `MaximumBatchSize`, `MultiplexingTimeout`, `Interval` and `MaxThreads` parameters for both the live and history aggregators.

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
  <sitecore>
    <aggregation>
      <!-- Configure the aggregator agent: -->
      <aggregator>
        <Aggregator>
          <MultiplexingTimeout>0.00:00:01</MultiplexingTimeout>
        </Aggregator>
        <MaximumBatchSize>128</MaximumBatchSize>
      </aggregator>
      <!-- Configure the historyWorker agent: -->
      <historyWorker>
        <Aggregator>
          <MultiplexingTimeout>0.00:00:01</MultiplexingTimeout>
        </Aggregator>
        <MaximumBatchSize>128</MaximumBatchSize>
      </historyWorker>
      <!-- Aggregation Module: -->
      <module type="Sitecore.Analytics.Aggregation.AggregationModule">
        <BackgroundServices>
          <aggregator>
            <Interval>0.00:00:15</Interval>
            <MaxThreads>1</MaxThreads>
          </aggregator>
          <history>
            <Interval>0.00:00:15</Interval>
            <MaxThreads>1</MaxThreads>
          </history>
        </BackgroundServices>
      </module>
  </sitecore>
</configuration>
```

Example 2: Enabling TVP support for a custom aggregation

This is a code sample that you can use to create your own configuration file to enable a custom aggregation to work with batch aggregation. It shows how to integrate a custom processor into the aggregation pipeline for interactions and how to configure mappings for a custom fact table with support for SQL Server TVP.

```
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
  <sitecore>
    <aggregation>
      <routines>

        <ExecRoutineStatementBuilder>
          <mappings>
            <!-- Reference to the fact table and associated stored procedure and table type -->
            <SqlMappingEntity uid="MyCustomFact"
               type="Sitecore.Analytics.Aggregation.SqlMappingEntity, Sitecore.Analytics.Sql">
              <Table>Fact_CustomFact</Table>
              <Routine>Add_CustomFact_Tvp</Routine>
              <IsMultiRow>True</IsMultiRow>
              <TableType>CustomFact_Type</TableType>
            </SqlMappingEntity>
          </mappings>
        </ExecRoutineStatementBuilder>
      </routines>
    </aggregation>
    <pipelines>
      <group groupName="analytics.aggregation">
        <pipelines>
          <interactions>
            <!-- Registration of the processor that will write to the new fact table -->
            <processor uid="MyCustomFact" type="Customization.CustomProcessor, Customization" />
          </interactions>
        </pipelines>
      </group>
    </pipelines>
  </sitecore>
</configuration>
```

Batch aggregation performance counters

If you want to adjust the performance of your batch aggregation processing configuration, or you want to monitor and troubleshoot the aggregation process, you can use the following counters to measure performance.

Counter	Description
Average batch size – live	Displays the average number of interactions contained in each batch processed during live aggregation.
Average batch size – history	Displays the average number of interactions contained in each batch processed during history aggregation. Under normal operation, this counter is equal to `MaxBatchSize` for most of the time while the history aggregation is active.
Number of batches containing failing items – live	Displays the number of live aggregation batches (per second) containing interactions that have failed to be aggregated or saved to the reporting database. Under normal operation, this counter is always zero.
Number of batches containing failing items – history	Displays the number of history aggregation batches (per second) containing interactions that have failed to be aggregated or saved to the reporting database. Under normal operation, this counter is always zero.
Average write time – live	Displays the average time (in milliseconds) that it takes during live aggregation for a batch to be saved to the reporting database. A number larger than the configured multiplexing timeout indicates a bottleneck.

Batch aggregation overview

In the Sitecore Experience Database (xDB), the aggregation process groups and reduces live or historical data from the MongoDB collection database so that it can be used by the SQL Server reporting database and Sitecore reporting applications. In a scalable xDB architecture, aggregation is usually performed on one or more dedicated aggregation servers.

When you configure a processing server for aggregation, you can specify the number of agents or threads that you want to run concurrently. Batch aggregation enables you to group interactions together into batches to improve the performance and throughput of your aggregation processing framework.

Batch processing:

- Makes more optimal use of SQL Server resources.
- Processes more interactions in fewer SQL Server transactions.
- Can improve the performance of your existing aggregation framework.
- Can reduce network traffic.
- Means that fewer database input/output operations are required to process interactions.

In xDB, batch aggregation comes as part of the standard Sitecore installation. The default number of interactions that you can process in a single batch has been set so that, for each transaction, the cost and execution time per row is low. However, solutions can vary, and you may need configure the batch aggregation settings to suit your own requirements.

You can change the default number of interactions that are processed in each batch in the `MaximumBatchSize` setting, and you can apply this setting separately for live or history collections.

Note

Do not make changes directly to the configuration files, but instead create your own custom configuration patch file that performs the required changes during run time.

Batch aggregation components

When you use batch aggregation, there are several components containing settings that you can change to improve the performance of your solution.

The batch aggregation agent

The batch aggregation agent is a background service that you can schedule to run at regular intervals to process live interactions. Each time it runs, it gathers a batch of interactions from the collection database and runs them through the batch aggregator. When the aggregator has finished, it marks each interaction that it has processed as complete and reschedules any interactions that have failed.

You can configure the batch aggregation agent using the `Sitecore.Analytics.Processing.Aggregation.Services.config` file.

The following example shows the default batch aggregation configuration:

```
<sitecore>
  <aggregation>
    <aggregator type="Sitecore.Analytics.Aggregation.InteractionBatchAggregationAgent,
       Sitecore.Analytics.Aggregation">
      <Context ref="aggregation/aggregationContexts/interaction/live" />
      <DateTimeStrategy ref="aggregation/dateTimePrecisionStrategy" />
      <Aggregator type="Sitecore.Analytics.Aggregation.InteractionBatchAggregator,
       Sitecore.Analytics.Aggregation" singleInstance="true">
        <MultiplexingTimeout>0.00:00:01</MultiplexingTimeout>
      </Aggregator>
      <MaximumBatchSize>64</MaximumBatchSize>
    </aggregator>
  <aggregation>
<sitecore>
```

You can change the following settings in the configuration file for the batch aggregation agent:

Configuration node	Description
`Context`	Specify the path or location of the data that you want to aggregate and the location where you want the results saved.
`Aggregator`	Specify the batch aggregator that you want to use to process interactions.
`MaximumBatchSize`	Specify the maximum number of interactions to include in a single batch.

The batch aggregator

The batch aggregator takes one or more interactions at a time, runs the aggregation pipeline for each interaction in the batch, and combines the aggregated data into a larger data set. The combined data set is then saved back to the reporting database.

The multiplexer

The multiplexer reduces the number of requests made to the reporting database by combining individual aggregation threads into a single batch or data set, which can then be saved more efficiently to the reporting database. This can significantly reduce the amount of traffic sent across the network.

The MultiplexingTimeout configuration setting enables you to specify the maximum time that you want the multiplexer to wait for other batch aggregators before saving the data set.

The Microsoft SQL Server reporting storage provider now supports storing batches of aggregation data sets, and its robustness has been improved. It has been optimized to save large data sets in a single transaction while at the same time minimizing the use of system resources.

The history worker

The history worker agent enables you to rebuild the reporting database, and it also comes with support for batch aggregation.

You can configure the history worker agent using the `Sitecore.Analytics.Processing.Aggregation.Services.config` file.

It contains the same parameters as the batch aggregation live agent: `MultiplexingTimeout` and `MaximumBatchSize`.

The following example shows the default configuration for the history worker:

```
<!-- Configure the historyWorker agent: -->
<historyWorker
 type="Sitecore.Analytics.Aggregation.Data.Processing.InteractionBatchHistoryWorker,
  Sitecore.Analytics.Aggregation">
  <HistoryTaskManager ref="aggregation/historyTaskManager" />
  <DateTimePrecisionStrategy ref="aggregation/dateTimePrecisionStrategy"/>
  <CollectionData ref="aggregation/collectionData" />
  <AggregationContext ref="aggregation/aggregationContexts/interaction/history" />
  <Aggregator type="Sitecore.Analytics.Aggregation.InteractionBatchAggregator,
   Sitecore.Analytics.Aggregation" singleInstance="true">
    <MultiplexingTimeout>0.00:00:01</MultiplexingTimeout>
  </Aggregator>
  <MaximumBatchSize>64</MaximumBatchSize>
</historyWorker>
```

Send feedback about the documentation to docsite@sitecore.net.

Clustered server environments

A clustered content delivery environment consists of a collection of servers grouped together to improve scalability and performance. Each cluster can contain two or more content delivery instances, each with its own session state server. You can also group clusters together in the same place or spread them across different geographical locations.

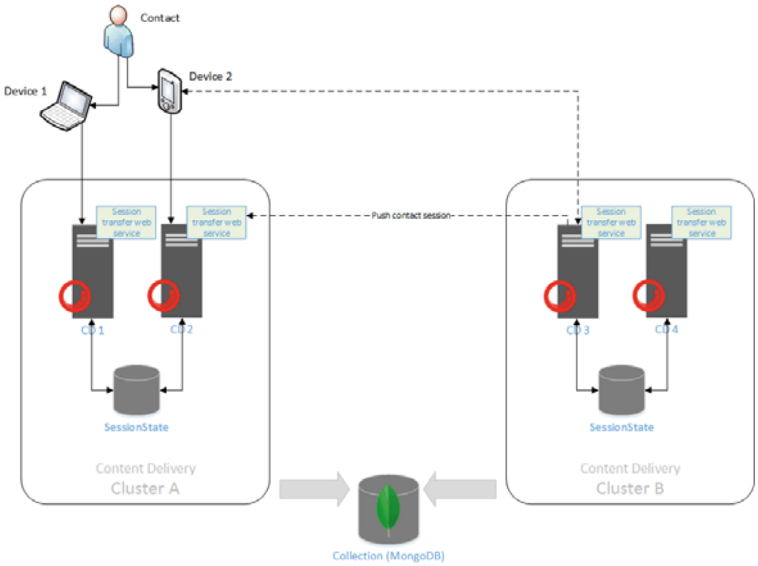
In a clustered content delivery environment, the same cluster of web servers serves all the visits for a single contact. This ensures that all interactions have fast access to the current state of the contact. If a single contact opens multiple concurrent sessions from different browsers or devices, the xDB ensures that each session sticks to the same cluster. A contact can only move to another cluster when they end all their current open sessions and start a new one or the session expires.

Content delivery server cluster architecture

You can configure a clustered server environment for content delivery and to improve scalability and performance. For example, to improve performance and enable horizontal scaling of content delivery, you can create multiple content delivery clusters.

A content delivery cluster consists of the following components:

- Two or more content delivery instances. Please refer to Configure a content delivery server for details on how to set up content delivery instances in a cluster.
- Session state database – to ensure that a contact sticks to the same content delivery cluster, each cluster must have a shared session state server (Microsoft SQL Server or MongoDB).
- Load balancer – that distributes traffic to the appropriate content delivery instances in the cluster (not shown in the diagram). The load balancer is the public endpoint of the cluster. The DNS name for the load balancer must use the same naming scheme that you use when you configure your clustered server environment. Two content delivery clusters with two content delivery instances in each cluster:



Server clusters and session state

Sitecore uses session state for storing information about the current contact interaction, contact state, and any other related data.

- Private session state collects data on the interactions and the devices used by the contact.
- Shared session state data includes all data that is unique to a contact and that can be shared across simultaneous sessions, such as contact details and any triggered engagement automation states.

If a contact opens multiple, concurrent sessions from different browsers or devices, the xDB ensures that each contact stays connected to the same server cluster where their interaction originated. A contact can only move from one cluster to another once the active sessions have ended or expired.

Server clusters and hostnames

Content delivery clusters share the same naming scheme, which is closely tied to the server geo-local DNS configuration settings:

- All server clusters respond to a global hostname that is configured in DNS to resolve to the IP address of the closest cluster. For example, *www.sitecore.net*, which maps to `4.4.4.4` in Asia and `8.8.8.8` in the United States.
- Each cluster must have a unique hostname that globally resolves to the cluster's IP address. For example, *asia.sitecore.net* maps to `4.4.4.4` globally, and *us.sitecore.net* maps to `8.8.8.8` globally.

Send feedback about the documentation to docsite@sitecore.net.

Live session agent

The Live Session Agent module enables you to install and configure an additional server role on a Sitecore content delivery instance. This additional server role enables you to trigger and process automation states by timeout, for contacts with live sessions on your website.

Note

Sitecore 8.0 and 8.1 supports *Sitecore Engagement Automation Live Session Agent 1.0* which is available to download from the [Sitecore Developer Portal](#).

For unlocked contacts, whose sessions have expired, engagement plans can still be executed by timeout but using the live session agent, engagement plans can be executed closer to the timeout delay set on the automation state.

Installation considerations:

- You should only install this module on Sitecore instances that are already part of a content delivery cluster.

  Note

  The Live Session Agent module must be installed on all instances within the cluster.

- Even though the module must be installed on all instances in a content delivery cluster, you can enable the background agent only on a subset of instances, such as on a dedicated instance or a set of instances that only process the timeout and not HTTP requests.
- To disable the background agent on a particular Sitecore instance, rename or remove the `Sitecore.EngagementAutomation.LiveSessionAgent.Processing.config` file in the `/App_Config/Include` folder.
- During [configuration](#), be aware that a server with this module installed can use significant additional resources in relation to:
  - Engagement automation plan execution when triggered by a timeout.
  - Supporting the Live Sessions Processing Agent.
  - Monitoring the queue of contacts that have engagement automation states with timeout.
- By installing and configuring the live session agent module on a dedicated content delivery instance, you create a more secure and scalable Sitecore solution, because the additional instance helps to boost the performance of timeout processing.
- The instance must also be connected to the same shared session state that is used by the content delivery cluster.

Send feedback about the documentation to docsite@sitecore.net.

Processing overview

The Sitecore Experience Database (xDB) uses two different types of databases:

- A MongoDB (NoSQL) collection database that collects all captured experience information in a loosely structured way
- An SQL Server reporting database that stores information extracted from the collection database in a form suitable for reporting

By default Sitecore xDB must always keep the reporting database in sync with the collection database.

Types of processing

Sitecore xDB has a data analysis layer that can perform several different kinds of processing on data stored in the collection database:

- Aggregation – aggregation processing extracts data from the collection database, then groups and reduces it before storing the data in the reporting database for use by Sitecore reporting applications.
- Continuous update – this process always keeps the reporting database in sync with the collection database.
- Rebuilding the reporting database – this process rebuilds the entire reporting database upon request.
- Maintenance – performs routine maintenance tasks on the collection database.

In daily operation the xDB uses a single reporting database that is continuously synchronized with new traffic information from the collection database.

Continuous update of the reporting database

Continuous update of the reporting database is the processing of interactions that have just ended. Unlike the rebuild of the reporting database, this is a continuous process that starts as soon as you launch Sitecore and which keeps the reporting database up to date with the most recent interactions, so long as the xDB is running.

The continuous update of the reporting database process works as follows:

1. The latest interactions are saved to the collection database.
2. Interaction data is added to the processing pool for aggregation.
3. An agent worker picks up the interaction from the processing pool and hands it on to an aggregator.
4. The aggregator pushes the interaction through the aggregation pipeline and converts the data into a form suitable for the reporting database.

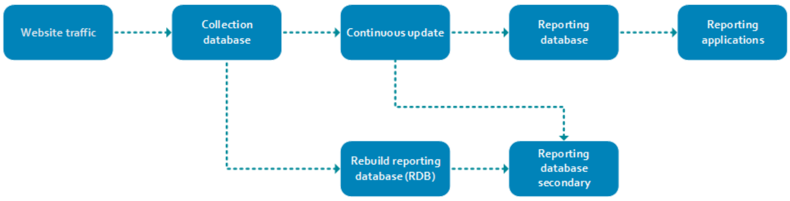
   The aggregation process converts data into a form that is easier to query and that is suitable for use with Sitecore reporting applications that use SQL Server.

   Once the data is converted into the correct format, it is merged into existing reporting data that is stored in the reporting database, keeping the reporting database continuously in sync with the latest interactions on your website.

Rebuilding the reporting database

Rebuild of the reporting database is the re-processing of interactions that have already been aggregated into the reporting database for use by Sitecore reporting applications. To ensure that the latest changes to the collection database appear in the reporting database, from time to time you need to rebuild the reporting database. When you rebuild the reporting database, its entire contents are overwritten.

This diagram shows the processing required to keep the reporting database continuously in sync with the collection database. This diagram shows the processing required to rebuild the entire reporting database on request.



In order to minimize interruptions to reporting functionality, the rebuild process works with a dedicated instance of the reporting database, called reporting secondary database (reporting.secondary).

When the rebuild process has finished, the reporting secondary database replaces the primary reporting database (reporting).

Sitecore xDB does not allow rebuilding of the primary reporting database in-place. So the reporting secondary database is only required during the rebuild process.

Note

You only need to connect a second reporting database if you intend to perform a rebuild of the reporting database.

The rebuild process is semi-automated but requires a system administrator to attach/replace databases in SQL Server and to make modifications to the xDB configuration.

Reasons for rebuilding the reporting database:

- After using the conversion tool to populate the reporting database (secondary) with analytics information from an earlier version of Sitecore.
- After having amended information in the collection database, in order to reflect the amendments in reports when looking at older data. An example of such an amendment could be assigning channels to referring sites.
- If you have reclassified a search key word or channel, aggregated report data is not updated automatically.
- If the reporting database has been lost or is irrecoverably out of sync with the collection database, for example, due to a disaster or if the details of two contacts have been merged.
- In Sitecore reporting applications it is possible to reclassify data that has already been processed by the aggregation layer. This could cause the reporting database to become out of sync with the collection database.

Important

When you rebuild the reporting database, the rebuild process overwrites all existing data in the reporting database.

Send feedback about the documentation to docsite@sitecore.net.

Reasons for rebuilding the reporting database

When you rebuild the reporting database, you reprocess the interactions that have already been aggregated into the reporting database.

The rebuilding process merges historical data from earlier versions of Sitecore with live xDB data making all the data available to Sitecore reporting applications. One of the main reasons for rebuilding the reporting database is as the final step of the conversion process.

Important

To ensure that the reporting database contains the latest data changes, you need to rebuild it from time to time. Remember that when you rebuild the reporting database, you overwrite its contents. For best results, you should use a clean copy of the `Sitecore.Analytics.mdf` database every time you perform a rebuild.

You should rebuild the reporting database if:

- You have used the conversion tool to populate the reporting database with analytics data from an earlier version of Sitecore.
- You have made changes to the collection database – a rebuild is necessary to reflect the changes in reports that use older data. For example, if you have changed the channel associated with one of your referring sites.
- You have reclassified reporting data – In Experience Analytics and other Sitecore reporting applications it is possible to reclassify data that has already been processed by the aggregation layer. For example, if you have reclassified a search key word or channel, aggregated report data is not automatically updated. This can cause the reporting database to become out of sync with the collection database.
- Your databases are out of sync – if the reporting database has been lost or has become out of sync with the collection database, for example, due to a disaster or if contacts have been merged. The accuracy of the reporting database can decrease over time as more contacts are merged and more interactions are imported. How often you need to rebuild the reporting database depends on your system architecture and the amount of merges and imports you need to process.

Note

The rebuild process is semi-automated but also requires an administrator to attach, detach, or replace databases in SQL Server, and also to modify some configuration files.

Send feedback about the documentation to docsite@sitecore.net.

Server configuration features

In the Sitecore Experience Platform, you can configure servers to perform dedicated tasks, such as content delivery, content management, reporting, and data processing. Configuring servers to perform different server roles allows you to begin the process of improving performance and scalability.

Note

To configure a server to perform a specific predefined feature or role, you may need to make changes to the default configuration of your Sitecore solution.

The following table is a description of the server features that you can enable or disable, along with the required connection strings and configuration files that you need to enable or disable to configure a role. The *App_Config* folder contains all connection strings, in the `ConnectionStrings.config` file, and it also contains all configuration files, in the *Include* folder.

Note

Do not make changes directly to the configuration files. Instead, you must create a patch file that performs the required changes during runtime.

Server feature and description	Connection strings	Configuration files

Tracking
 	
To track online visits, contacts (visitors), personalization, goals, campaigns, profiles, pattern cards, and multivariate tests.
 	
By default, all tracking data is saved in the MongoDB collection database.
	Configure the following database connection strings:
	
	`analytics`
	
	`tracking.live`
	
	`tracking.history`
		Required configuration files:
		
		`Sitecore.Analytics.Tracking.config`
		
		`Sitecore.Analytics.Tracking.Database.config`
		
		`Sitecore.Analytics.Tracking.Aggregation.config`
		
		`Sitecore.Analytics.Tracking.RobotDetection.config`
		
		If you want all tracking subsystem features enabled but you do not want tracking data to be saved to the collection database, you need to disable the following configuration files:
		
		`Sitecore.Analytics.Tracking.Database.config`
		
		`Sitecore.Analytics.Tracking.Aggregation.config`
		
		`Sitecore.Analytics.Tracking.RobotDetection.config`

Reporting
 	
Reporting uses data from the collection and reporting databases for applications such as Experience Analytics reports.
	Configure the following database connection strings:
	
	`analytics`
	
	`reporting`
		Required configuration file:
		
		`Sitecore.Analytics.Reporting.config`

Processing
 	
In the xDB, processing can perform a variety of distributed tasks via the *TaskManager* API. For example, location and contact reclassification and the rebuilding of the reporting database.
 	
You can define and start your own distributed processing tasks using the *TaskManager* API. It is only possible to run custom tasks if you have enabled the processing feature on at least one server.
	Configure the following database connection strings:
	
	`tracking.live`
	
	`tracking.history`
	
	`analytics`
	
	`reporting.secondary`
		Required configuration file:
		
		`Sitecore.Processing.config`
		
		`Sitecore.Analytics.Processing.config`
		
		`Sitecore.Analytics.Processing.Services.config`

Aggregation
 	
Aggregation processing involves the extraction of data from the collection database, grouping and reducing the data, and then storing it in the reporting database for use by Sitecore reporting applications. Aggregation keeps the reporting database synchronized with the collection database.
	Configure the following database connection strings:
	
	`tracking.contact`
	
	`tracking.live`
	
	`reporting`
		Required configuration files:
		
		`Sitecore.Analytics.Processing.Aggregation.config`
		
		`Sitecore.Analytics.Processing.Aggregation.ProcessingPools.config`
		
		`Sitecore.Analytics.Processing.Aggregation.Services.config`

Reporting Service Client
 	
Enable the reporting client feature to view Sitecore reporting applications. Enable this feature on a content management server.
 	
If you have the appropriate network/security setup, such as in the xDB cloud, the reporting UI can display reports even if the reporting database is not available.
	Configure the following database connection strings:
	
	`analytics`
		Required configuration files:
		
		`Sitecore.Analytics.Reporting.config`
		
		`Sitecore.Xdb.Remote.Client.config`

Reporting Service Server
 	
Can query different data sources for Sitecore reporting applications. It queries the reporting database for grouped, aggregated data and trends, and the collection database for individual contact or interaction data.
	Configure the following database connection strings:
	
	`reporting`
		Required configuration files:
		
		`Sitecore.Analytics.Reporting.config`
		
		`Sitecore.Xdb.Remote.Server.config`

EAS Processing Processing of engagement automation state timeouts. This type of processing sets up the automation workers that process EAS timeout conditions.	Configure the following database connection strings: `analytics`	Required configuration file: `Sitecore.Analytics.Automation.TimeoutProcessing.config`

Note

The following configuration files are necessary for all server features and you should therefore not disable them:

`Sitecore.Analytics.config`

`Sitecore.Analytics.Model.config`

`Sitecore.Analytics.Mongodb.config`

By default, all of these configuration files are enabled with the exception of `Sitecore.Xdb.Reporting.RemoteClient.config` and `Sitecore.Xdb.Reporting.RemoteServer.config` file, which are disabled by default.

Send feedback about the documentation to docsite@sitecore.net.

Walkthrough: Rebuilding the reporting database

The Sitecore Experience Database (xDB) uses two database systems (SQL Server and MongoDB) that need to be kept in sync to provide accurate data for Sitecore reporting applications:

- MongoDB collection database - stores all experience data
- SQL Server reporting database - stores aggregated reporting data

Sitecore automatically keeps these databases in sync but there are certain circumstances when you may need to perform a complete rebuild of the reporting database. For example, after you have run the Sitecore Analytics Conversion Tool and have additional historical data that you want to include in your reporting applications. There are also several other reasons for rebuilding the reporting database.

When you rebuild the reporting database, to avoid disruption during the rebuild process, you connect a secondary reporting database to store all aggregated data as it is reprocessed.

This walkthrough describes how to:

- Check prerequisite databases
- Create and connect a second reporting database
- Change the time to clear storage setting
- Run the Rebuild Reporting Database page
- Reconfigure reporting connection strings

Check prerequisite databases

Before you start the rebuild process, first check that you have the MongoDB analytics database and the SQL Server reporting database connected. These databases are part of a standard Sitecore installation. You only need to create and configure a secondary reporting database if you intend perform the rebuild process.

Each database has a separate connection string that you can configure in the `App_Config/ConnectionStrings.config` file:

- Collection database (MongoDB) – can contain historical data from a converted Sitecore DMS SQL Server database and new data collected from your website (live data).

  MongoDB collection database connection string:

  ```
  <add name="analytics" connectionString="mongodb://localhost/analytics"/>
  ```

- Primary reporting database (SQL Server) – contains all the live data collected on your website after upgrading to a newer version of Sitecore.

  Primary reporting database connection string:

  ```
  <add name="reporting" connectionString="user
  id=_sql_server_user_;password=_user_password_;Data
  Source=_sqlserver_;Database=Sitecore_Reporting" />
  ```

- Secondary reporting database (SQL Server) stores historical data during the rebuilding of the reporting database. See the next section for detailed instructions on how to create and connect a secondary reporting database.

  Secondary reporting database connection string:

  ```
  <add name="reporting.secondary" connectionString="user
  id=_sql_server_user_;password=_user_password_;Data
  Source=_sqlserver_;Database= Sitecore_Reporting_Secondary" />
  ```

Create and connect a second reporting database

The first step in the rebuild process is to create and then connect a secondary reporting database.

To connect and configure a secondary reporting database:

1. Take a clean copy of the `Sitecore_Analytics` database from your Sitecore distribution to use as your secondary reporting database. For best results, always use a clean copy.
2. In SQL Server Management Studio, attach the copied database to your SQL Server instance. Use the name `reporting.secondary` in the connection string.

   Note

   The reporting and secondary reporting databases can take up significant disk space so you may need to plan for extra storage requirements.

3. Add or edit the `reporting.secondary` connection string to point to the newly created database. For example:

   ```
   <add name="reporting.secondary" connectionString="user
   id=_sql_server_user_;password=_user_password_;Data
   Source=_sqlserver_;Database=Sitecore_Reporting_Secondary" />
   ```

   Note

   If you have installed the Sitecore upgrade package, you may have already added a secondary reporting database, and if so, you do not need to perform this step twice.

4. If you are configuring a dedicated server, check that you have enabled the history processing pool in the appropriate configuration file.

   For example, to enable history processing on a dedicated content delivery server, navigate to the following processing configuration file:

   ```
   App_Config/Include/Sitecore.Analytics.Processing.Aggregation.ProcessingPools.config
   ```

   In this config file, make sure the `<Enabled>` parameter is set to true: `/configuration/sitecore/aggregationProcessing/processingPools/history/Enabled`

   ```
   <history type="Sitecore.Analytics.Data.MongoDb.ProcessingPool.MongoDbProcessingPool,
     Sitecore.Analytics.MongoDb" singleInstance="true" >
     <param desc="connectionStringName">tracking.history</param>
     <Name>history</Name>
     <Enabled>true</Enabled>
   </history>
   ```

Change the time to clear storage setting

In the latest version of Sitecore xDB, the primary SQL Server reporting database contains some additional marketing definition tables that you need to copy to the secondary reporting database. When you run the rebuild reporting database page ensure that you allow more time for the clear storage process while these tables are being copied.

The default time to clear storage setting is 1 minute. Change this setting to a time interval appropriate for your Sitecore solution, for example, 10 minutes.

To change the `TimeToClearStorage` setting:

1. Open the `Sitecore.Analytics.Processing.Aggregation.config` file.
2. Change the `TimeToClearStorage` setting to an appropriate time. For example 10 minutes.

```
<reportingStorageManager type="Sitecore.Analytics.Aggregation.History.ReportingStorageManager" singleInstance="true">
  <TimeToClearStorage>0.00:01:00</TimeToClearStorage>
```

Run the Rebuild Reporting Database page

To rebuild the reporting database:

1. In a web browser window, open the rebuild reporting database history processing page using the following path:

   ```
   <sitename>/sitecore/admin/RebuildReportingDB.aspx
   ```

   Note

   You can only run the `RebuildReportingDB.aspx` and `RedeployMarketingData.aspx` pages from a content management server.

2. Click Start to begin rebuilding the reporting database (synchronization processing).
3. In the *Rebuild Reporting Database* page, when you see *Waiting to receive to data* status, copy the following marketing definition tables from the primary to the secondary reporting database:
   - `CampaignActivityDefinitions`
   - `GoalDefinitions`
   - `OutcomeDefinitions`
   - `MarketingAssetDefinitions`
   - `Taxonomy_TaxonEntity`
   - `Taxonomy_TaxonEntityFieldDefinition`
   - `Taxonomy_TaxonEntityFieldValue`

   Important

   It is important that you add these additional tables to the secondary reporting database while the *Waiting to receive to data* status is displayed in *Rebuild Reporting Database* page. Ensure that you have also changed the `TimeToClearStorage` setting in the `Sitecore.Analytics.Processing.Aggregation.config` file to allow sufficient time for these tables to be added.

   After you have added these tables the *Rebuild Reporting Database* page resumes the rebuilding process.

4. The rebuild reporting database tool provides feedback while it is processing until the rebuild process is completed.

Note

When you run the rebuild process (synchronization), this erases all information in the `Sitecore_Reporting_Secondary` database. Therefore, you should always use a clean copy of the database when you rebuild the reporting database because this speeds up the rebuild process and uses fewer resources.

Reconfigure reporting connections strings

After the rebuild process has finished, you need to reconfigure the reporting and reporting secondary connection strings:

1. Navigate to the `Website/App_Config` folder, and change the names in the `ConnectionStrings.config` file `Database=` attribute, so that the reporting connection string points to the *Sitecore_Reporting_Secondary* database (this has replaced the *Sitecore_Reporting* database).
2. Comment out the `reporting.secondary` connection string.

   The following samples show the connection strings after swapping the database names and commenting out the `reporting.secondary` connection string:

   ```
   <add name="reporting" connectionString="user
   id=_sql_server_user_;password=_user_password_;Data
   Source=_sqlserver_;Database=Sitecore_Reporting_Secondary" />
   <!--
   <add name="reporting.secondary" connectionString="user
   id=_sql_server_user_;password=_user_password_;Data
   Source=_sqlserver_;Database=Sitecore_Reporting2" />
   -->
   ```

   The xDB now uses the *Sitecore_Reporting_Secondary* database as its primary reporting database to collect new, live data from your website. The *Sitecore_Reporting2* database is disabled until you need to enable it again the next time you rebuild the reporting database.

   Note

   Sitecore_Reporting2 is an example name that you can use for the clean reporting database next time you rebuild the reporting database.

3. To verify that the rebuild has been successful, open the Experience Analytics reporting application and view reports that show the most recent visits to your website. The visits shown in these reports should correspond to the historical data that came from the MongoDB collection database.

   Note

   If you encounter problems, you can easily swap back the connection strings to restore the previous version of the reporting database.

Send feedback about the documentation to docsite@sitecore.net.

Configuring session database servers

In the xDB, you configure a session state database to track all contact interactions and to maintain visitor session state. Configuring out of process session state is particularly important if you have deployed an environment with multiple content delivery servers.

You can configure a session state database on the same server as other roles, such as content delivery, but for better performance and scalability, create a separate session database server.

Depending on which database system you want to use when you configure your session state database, you need to use a session state provider. Sitecore comes with two providers:

- *Sitecore ASP.NET Session State Provider for MongoDB*
- *Sitecore ASP.NET Session State Store Provider for Microsoft SQL Server*

Note

A dedicated Sitecore instance is not required when configuring a dedicated session state database server.

You can choose to use the same database for both session states but to ensure that the system can distinguish between the private and shared session entries, you must remember to configure the correct attribute in the web.config and the Sitecore.Analytics.Tracking.config files.

Follow these instructions to configure session state providers:

- Walkthrough: Configuring a private session state database using the MongoDB provider
- Walkthrough: Configuring a private session state database using the SQL Server provider
- Walkthrough: Configuring a shared session state database using the MongoDB provider
- Walkthrough: Configure a shared session state database using the SQL Server provider

Send feedback about the documentation to docsite@sitecore.net.

Estimate memory and storage requirements for session state

In order to plan your content delivery server and session state database server deployments, you need to estimate your memory and disk space capacity requirements for your session state store.

Depending on your server configuration, your session state store can use memory or a physical session database before saving session data to the xDB at the end of each web session.

When estimating your storage requirements for a session state store, consider the following:

- Private session state size – typically, a clean Sitecore installation needs approximately 30 kb of storage capacity per session.
- Shared session state size – space requirements depend on the number of contacts you have in xDB and how many customizations you have implemented. For example, you need more space if you have custom contact facets populated with data or if you want to load a contact's visit history for personalization. To estimate the number of contacts, you can query the database where your contacts are stored.
- Session duration – in xDB, you can analyze interactions to find out the average duration of web sessions on your website. Shared sessions tend to be longer than private sessions because multiple visits from the same contact overlap in the shared session state object. For a typical website, the number of concurrent shared sessions is equal to or slightly less than the number of concurrent private sessions.

To calculate your storage requirements, use this formula:

- Requests per second/average page views per session * (session duration in seconds + session end overhead) * estimated session size.

Metric	Description
Requests per second	The average number of HTTP requests that the website serves every second. You can get this value from the ASP.NET performance counters.
Average page views per session	The number of HTTP requests that makes up an average website session. You can get this information from the xDB.
Session duration in seconds	The amount of time that the session is present in the session database. This calculation is based on: How long the visitor is active on the website – the difference between the `StartDateTime` and `EndDateTime` in the interaction record. Session timeout – the session timeout value defined in your ASP.NET configuration. Session end overhead – approximately one minute.

Session end overhead	The average time it takes for Sitecore to process an expired session and to remove it from the session state store. This value is calculated in seconds and does not typically exceed one minute.

Example

To calculate the storage requirements for the following website requirements:

- Session timeout: 20 minutes
- Session end overhead: approx.1 minute
- Average session: 7 page views
- Average visit: 5 minutes
- Registered traffic: 10 requests per second

Estimated storage requirement calculation:

- 10 RPS / 7 * (5 + 20 + 1) * 60 * 30k = approximately 64MB.

Send feedback about the documentation to docsite@sitecore.net.

MongoDB session database performance considerations

For each web request, the content delivery server accesses the session-state store database multiple times. This can have a significant impact on the performance of your website. Therefore, you should use solid-state drives for the database and install enough memory to avoid frequent disk reads.

If you are using a MongoDB database as your session-state store, you should think about the following hardware and performance considerations.

Each visit requires about 30 kilobytes of storage capacity in the session database (by default) but space requirements may change depending on customizations.

You can calculate the disk space requirements expressed in kilobytes using the following formula:

- Maximum number of concurrent visits * 30

Note

This includes active visits and inactive visits that have not timed out.

You can also use the same formula to calculate memory requirements.

The MongoDB website has more information on how to calculate your RAM requirements and also on installation steps:

- MongoDB RAM
- MongoDB storage
- MongoDB installation guide for Windows

Send feedback about the documentation to docsite@sitecore.net.

Session state

In the Sitecore Experience Database, you can use a session state server to share all your contact sessions across different browsers and devices. Configuring out-of-process session state is important if you have deployed an environment with multiple content delivery servers or clusters of content delivery servers.

Session state overview

Session state is a way of identifying contacts on your website by assigning them a unique session ID and by using cookies. Every time a contact makes a request, such as triggering a campaign or a goal this information is held in session state until the end of session.

This means that fewer server trips are needed to write data back and forth to the collection database. Instead, xDB makes one trip at the start of a session to identify the contact and a second at the end to save all the session data back to the collection database.

Tracking contact session state is important regardless of whether you install a standalone environment or a fully scalable environment. For example, it is possible to track two simultaneous contact sessions from two different devices, such as from a desktop web browser or a mobile phone.

If you decide to deploy a large, fully scalable environment with vertical and horizontal scaling, then session state becomes even more important. For example, in a multi-content delivery cluster environment, you can share contact information between content delivery instances to ensure that contacts stay connected to the particular cluster where their interaction originated.

Note

You should only configure one session state server for each content delivery cluster.

In xDB, there are two types of session states:

- Private
- Shared

Important

You must configure both private and shared session states. You can use the same session provider for both private and shared sessions but to ensure that the system can distinguish between the two entries you must configure the `sessionType` attribute in the `web.config` and `Sitecore.Analytics.Tracking.config` with the correct attribute name, which can be *shared* or *private*.

Private session state

In private session state (ASP.NET), all analytics session data related to a specific interaction is collected and saved to the session state server.

The private session data collected during a session includes data on the interaction, and the devices used by a contact.

The default provider is the *in process* (inProc) session provider which uses session state stored in-memory. This is suitable when there is only a single content delivery server or multiple content delivery servers with sticky load balancing configured.

If you have multiple content delivery servers with a non-sticky load balancer configured, you should use one of the following *out of process* ASP.NET session state providers to share the private session state data between load balanced content delivery instances:

- Sitecore ASP.NET Session State Store Provider for MongoDB
- Sitecore ASP.NET Session State Store Provider for Microsoft SQL Server

These providers enable you to choose whether you want to use the MongoDB or SQL Server session database as your session state store. They also support the session end event that is required by the xDB to track website visits reliably. You could also choose an external session state provider so long as it supports the `Session_end` event.

The standard ASP.NET session state stores a cookie in the web browser. This means that if a contact uses Sitecore from two different devices, or even two different browsers on the same device, two separate sessions are created.

Note

Private session state is not required to support the `Session_End` event on content management servers. Private session state must support the `Session_End` event on content delivery servers.Out of process private session state is supported only on content delivery servers.

Note

You should only configure one session state server for each content delivery cluster that you have.

Shared session state

Contact data is collected and stored in a dedicated Shared session state store. This works in the same way as ASP.NET session state.

The default provider is the in process (inProc) session provider which uses session state stored in-memory. When there is only a single content delivery server, shared session state can be stored in memory because there is no need to share the session state with other servers. In this case, shared session state is still useful for handling concurrent visits by the same contact from different devices.

If you have multiple content delivery servers, you should choose the Redis or the SQL Server session provider to support *out of process* shared session state.

Shared session state data includes all data that is unique to a contact and that can be shared across simultaneous sessions, such as contact details, devices and engagement automation states triggered. This information needs to remain accessible to other concurrent operations, for example, interactions and background processes, and can be shared across multiple sessions.

Sitecore uses shared session state for storing information about the current contact state and related data.

In contrast to ASP.NET session state (which is exclusive to an interaction), shared session state stores a contact's session and is accessible from any visit session that is associated with the contact within the same cluster.

Note

Shared session state must support the `SessionEnd` event on content delivery servers. Out of process shared session state is supported only on content delivery servers.

Note

The standard `OutOfProcSessionStateStore` that ships with ASP.NET does not support the `SessionEnd` event and therefore cannot be used with shared session state.

Session Lifetime (Shared)

The following graph shows the lifetime of the sessions for a single contact and two concurrent interactions made from two different devices.


Picture 12

Session state providers

A session state server stores information relevant to current contact sessions. The Sitecore xDB comes with two session state services:

- Private session state holds information private to sessions. This is contact visit information, such as pages viewed, goals converted, or campaigns triggered.
- Shared session state holds information that may be shared by multiple visits on the same cluster, such as contacts and devices.

You can configure session state in two different ways: as either *in process* (InProc) or *out of process*.

In process

In process (InProc) is the default session state provider that comes with the Microsoft .NET Framework. It uses internal memory to store session data.

In process is the most suitable way of handling private session state for all data related to a specific interaction (single visitor session or visit). It is the only supported mode to use for content management servers.

Note

You cannot use InProc for private session state in a load-balanced environment unless you configure the load balancer to use sticky sessions.

Out of process

Out of process session state means that you use an external session state provider such as MongoDB or SQL Server. You can also use any custom provider that supports the `Session_End` event. Out of process is suitable for handling shared or private session state if you have multiple content delivery servers.

Important

Always use *out of process* session state if you have more than one content delivery instance in a cluster.

Sitecore comes with the following two session state providers for configuring *out of process* session state:

- Sitecore ASP.NET Session State Provider for MongoDB
- Sitecore ASP.NET Session State Store Provider for Microsoft SQL Server

When you configure session state on different types of Sitecore server configurations there are several different session state configuration scenarios you can follow.

Send feedback about the documentation to docsite@sitecore.net.

Session state configuration scenarios

In the Sitecore Experience Database, you can use a session state server to track all your contact sessions across different browsers and devices. Configuring session state is particularly important if you have deployed an environment with multiple content delivery servers or clusters of content delivery servers.

The following example scenarios provide guidance on how to configure session state on different types of Sitecore server configurations.

Single standalone server

On a single, standalone Sitecore instance, both private and shared session providers should use the in process (InProc) session state mode.

Single content delivery server and a separate content management server

On a content delivery server and on a content mangement server, you should configure both private and shared session providers to use the in process (InProc) session state mode.

Important

When configuring session state on content management servers, you should always configure session state providers to use the in process (InProc) mode. Sitecore does not support sharing session state among a group of content management servers or between content management and content delivery servers. Out of process session state is only supported on content delivery servers.

Content delivery cluster with a sticky load balancer

On all content delivery instances, you should configure the private (ASP.NET) session state provider to use the in process (InProc) mode. You should configure the shared session state provider to use one of the out of process modes connected to a database shared among all the content delivery instances.

Content delivery cluster with a non-sticky load balancer

On all content delivery instances, you should configure both session state providers to use one of the out of process modes connected to a database (or databases) shared among the content delivery instances.

Send feedback about the documentation to docsite@sitecore.net.