Web Forms for Marketers 8.0

All the official Sitecore documentation.

sitecore®
Own the experience™

Add an ASCX control to the page

In the Web Forms for Marketers module, you can convert and export a form to an .ascx file and then add it to your website as an ASCX control. For developers, this can make it easier to develop their custom form control.

To add an ASCX control to the page:

1. Using a text editor, in the `\layouts` folder of your Sitecore installation, create a new `default.aspx` page and insert the following code:

```
<%@ Page Language="C#" AutoEventWireup="true" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    </form>
</body>
</html>
```

2. In the Form Designer, in the Export group, click To ASCX and export the relevant form data to the `\layouts` folder. Use the `forms.ascx` format for the name of the file.
3. In the `default.aspx` page that you created, add the following code after the first `<%@ Page>` tag:

```
<%@ Register Src="form.ascx" TagName="SimpleForm" TagPrefix="uc1" %>
```

4. Inside the `<form>` tag, insert the following code:

```
<div>
   <uc1:SimpleForm ID="WebUserControl1" runat="server" />
</div>
</form>
```

5. In your web browser, enter the following URL: http://localhost/layouts/default.aspx.

Note

After you make changes to the .ascx file, the module does not verify that the form works correctly

Change the color and style of a web form

You can change the default color scheme and theme of web forms and MVC forms, for example, if you want your web forms to follow the style of your website. A color scheme sets the colors for each form element, such as the foreground or background color, and themes are a set of visual effects.

CSS classes describe colors and effects for an individual form field. You can change the style directly in the Form Designer using a predefined form field, or create your own custom CSS theme and style.

Note

Changes made to the theme and color of one form apply to all forms.

This topic describes how to:

Change the theme or color of web forms

To change the theme or color of web forms:

1. In the content tree, navigate to and select the folder where your web forms are stored.

   For example: (*/sitecore/System/Modules/Web Forms for Marketers/Website)*.

   Note

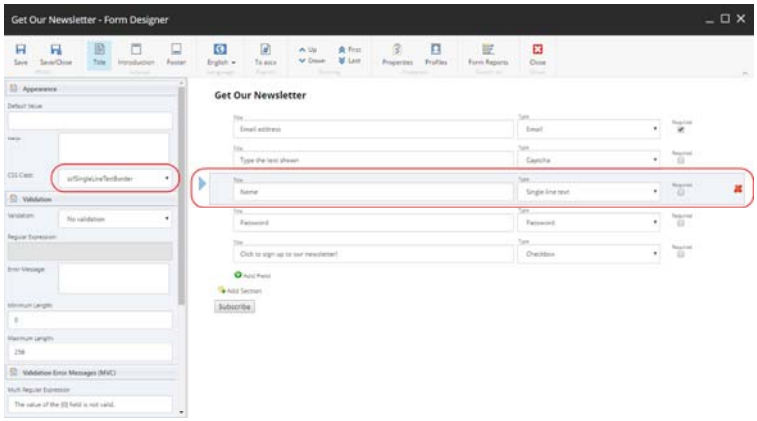
   The location of this folder may be different depending on your Sitecore configuration.

2. In the right pane, in the Themes section, click the Theme or Color drop-down list and select the theme or color that you want.
3. Click Save, and republish your website.

Change the style of a form field

To change the style of a form field:

1. In the Form Designer, in the relevant form, in the right pane, click the form field that you want to change the style of, for example, *Name*.



2. In the left pane, in the Appearance section, in the *Css Class* property, click the drop-down arrow and select a class.
3. Click Save or Save/Close and republish your website.

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

Create a custom CSS style in a web form

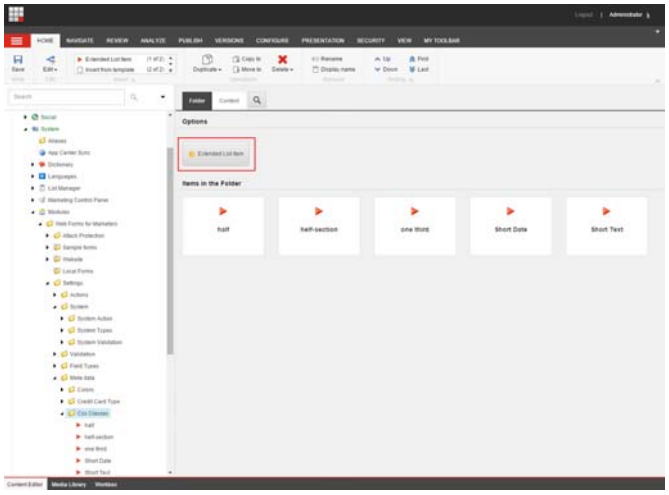
The Web Forms for Marketers module contains CSS styles that you can apply to form fields. Every field type has a default CSS style that is applied to it. You can also create custom CSS styles.

To create a custom CSS style:

1. In the `Website\sitecore modules\Shell\Web Forms for Marketers\Themes\` folder, in the `custom.css` file, define a new CSS style.

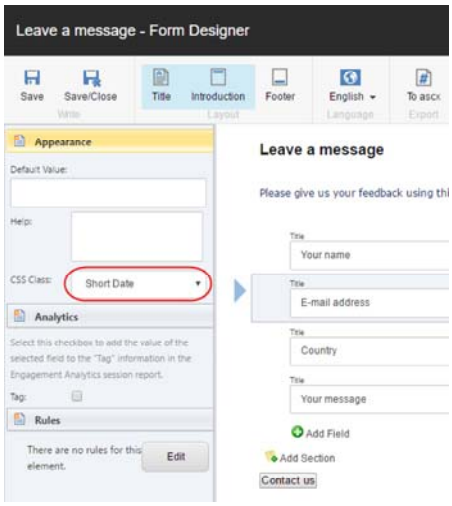
```
.scfShortDate
{
    clear: left;
    text-align: left;
    display: block;
    margin: 5px 0px;
    vertical-align: top;
    width: 60%;
}
```

2. In the Content Editor, navigate to the *sitecore/System/Modules/Web Forms for Marketers/Settings/Meta data/CSS Classes* folder, and in the right pane click Extended List Item.



3. In the Message dialog box, enter the name of the CSS style, for example *Short Date*, and then click OK.
4. In the right pane, in the Value field, enter the name of the CSS style. Click Save.

You can now apply your custom CSS style to any form field in the Form Designer.



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

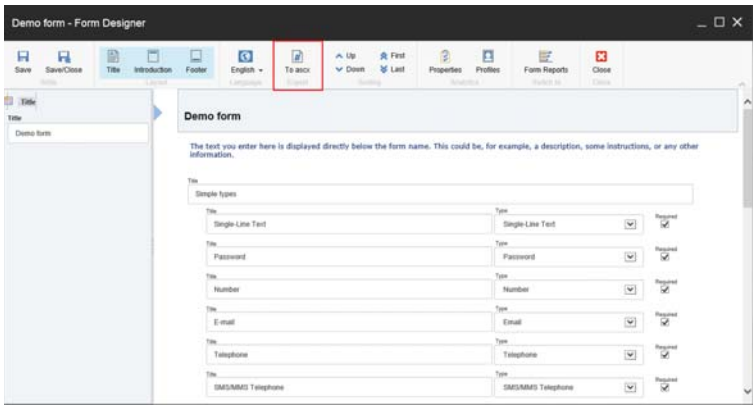
Export a web form to ASCX

Converting a web form to an .ascx file is good practice when you want to change the behavior or the appearance of a particular web form without affecting all other web forms.

Convert a web form to a .ascx file

To convert a web form to an `.ascx` file:

1. Open the Form Designer and in the Select a Form dialog, select the form you would like to export. Click OK.
2. In the Form Designer, in the Export group, click To ascx to convert a form to an `.ascx` file. The code is displayed in the Export to .ascx file dialog.



3. In the Export to .ascx file dialog, click Download to download the code as an `.ascx` file.

Note

After you have converted a web form to an `.ascx` file, do not delete the web form item in the Content Tree. This item is required for the analytics reports.

You can use converting a web form to an `.ascx` file for showing or hiding a field in a web form depending on another field value.

Multisite implementation with web forms

The Web Forms for Marketers module supports multisite environments. This means that administrators can define different form locations and settings for different websites. You can do this in the `formsRoot` attribute in the definition of the site in the `.config` files.

The value of this attribute is a Sitecore path that defines:

- The folder that stores the forms in the current website.
- The appearance and color settings for forms in the current website.
- Access rights.

The `formsRoot` parameter must contain either an item path or the ID of the target item. The target item must be based on the */sitecore/Templates/Web Forms for Marketers/Forms Folder* template.

For example, this is how the `formsRoot` parameter is defined in the `web.config` file:

```
<sites>
<site
name="samplesite"
virtualFolder="/"
physicalFolder="/"
rootPath="/sitecore/content"
startItem="/forms" database="web" domain="extranet"
formsRoot="/sitecore/System/modules/Web Forms for Marketers/local forms"
...
This can be defined in the Sitecore.Forms.config file using the ID:
<site name="website">
<patch:attribute name="formsRoot">
{F1F7AAB6-C8CE-422F-A214-F610C109FA63}
</patch:attribute>
</site>
</sites>
```

To ensure that there are no duplicated values, you should define the `formsRoot` parameter in the `\App_Config\Include\Sitecore.Forms.config` file.

When the `formsRoot` attribute is not defined for a website, new forms are stored in the content tree, in the */sitecore/System/Modules/Web Forms for Marketers/Local Forms* folder.

Reinstall the Web Forms for Marketers module

You can reinstall the Web Forms for Marketers module to make sure all the files and items are correct. The files and items can become corrupt during an unsuccessful upgrade.

Note

When you reinstall the module, all the save action parameters are reset to their default values.

To reinstall the module and save all of the created web forms:

1. Create a Sitecore package that contains all the web forms that you have created.
2. Back up the module database.
3. If you modified the `Sitecore.Forms.config` file, back up this file.
4. Install the Web Forms for Marketers installation package.
    - Click Overwrite all when prompted.
    - Click Continue always when prompted.
5. When you finish the installation, install a package with web form items.
6. Restore the module database.
7. Restore the `Sitecore.Forms.config` file.
8. Now you must configure the save action parameters because they have been reset to default values. For example, in the *Send Email Message* save action, set the host parameter.

Run web forms in live mode

In Sitecore, you can run a website directly from the Master database – this is referred to as *running in live mode*. Running in live mode eliminates the need to publish content and is similar to viewing a website in the Preview client.

A website that is configured to run in live mode functions exactly like a normal website. Live mode respects all publishing restrictions and workflows in the same way that a default website supports these features.

To run the Web Forms for Marketers module in live mode, edit the `web.config` file:

1. In the relevant `<site>` section, change `database="web"` to `database="master"`.
2. In the `<modules_website>` section, change `database="web"` to `database="master"`.

   The `<sites>` section of your `web.config` file should look like this:

```
<sites>
...
      <site name="modules_website"...database="master".../>
      <site name="website"...database="master".../>
...
</sites>
```

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

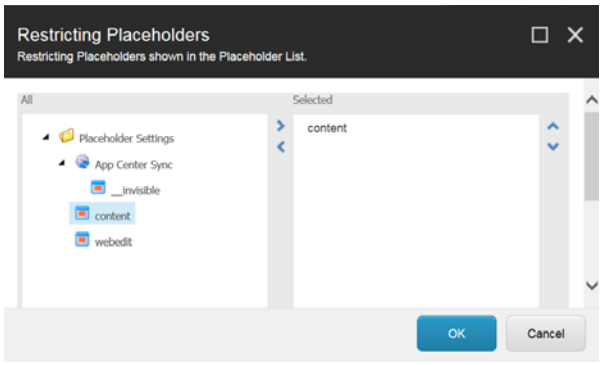
Select placeholders shown in the Placeholder List

A placeholder is an object in Sitecore that serves as an anchor point for other presentation objects and they allow content blocks to be added in predefined positions. Every placeholder is named using its "key" attribute. Renderings and sub-layouts can be associated with placeholders based on the placeholder key.

The Insert a New Form wizard only allows you to add web forms to placeholders that have *Placeholder Settings* items. A user who adds a new form must have write access to an item where the form is added, in order to see placeholders in the Placeholder List. This enables developers and website administrators to define which placeholders can contain a form.

The Restricting Placeholders wizard helps you select the list of placeholders that are shown in the Placeholder List of the Insert a New Form wizard.

1. To open the Restricting Placeholders wizard, on the Sitecore Desktop, click the Sitecore Start button, All Applications, Web Form for Marketers, and then click Restricting Placeholders.



   The Selected field, lists the placeholders that users can add a new form to.

2. To add a placeholder from the All section to the Selected section, select a placeholder and click ›.
3. To remove a placeholder from the Selected section, click ‹. When you click OK, all the changes are saved.

   Note

   To add a placeholder to the All section in the Restricting Placeholders dialog, you must create a new placeholder under the *Sitecore/Layout/Placeholder Settings* item in the content tree.

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

The security roles in the Web Forms for Marketers module

To configure the access that a user has to the features in the Web Forms for Marketers module, you can assign the necessary access rights through the security roles described in this topic.

To give access to all the features in the module, you can assign the following roles to a user:

- *Sitecore Marketer Form Author*
- *Sitecore Client Developing*
- *Sitecore Client Securing*

The following roles are relevant to Web Forms for Marketers module users:

Role	Description

Sitecore Client Forms Author	Allows the user to: • Create a new form. • Edit an existing form. • View the Form Reports.
Sitecore Client Developing	Allows the user to convert a web form to an .ascx file, in order to change the behavior or appearance of a web form.
Analytics Maintaining	Allows the user to: • Enable or disable form dropout tracking and associate goals with web forms. • Associate a new profile card with an item. • Access reports of Sitecore Engagement Analytics (SEA)
Sitecore Marketer Form Author	Inherits access rights from the following roles: • *Sitecore Client Forms Author* • *Analytics Maintaining* • *Analytics Reporting*
Sitecore Client Securing	Allows the user to: • Edit the *Create User* save action. • Edit the *Edit Role Membership* save action. • Edit the *Change Password* save action.

Uninstall the Web Forms for Marketers module

The Web Forms for Marketers module can be uninstalled by removing all the related files and items from Sitecore. You can also reinstall the module, if necessary.

To uninstall the Web Forms for Marketers module:

1. In the Master database, delete the following items:
   - */sitecore/layout/Renderings/Modules/Web Forms for Marketers*
   - */sitecore/Media Library/Web Forms for Marketers*
   - */sitecore/System/Modules/Web Forms for Marketers*
   - */sitecore/System/Marketing Center/Goals/Leave a message*
   - */sitecore/System/Marketing Center/Goals/Tell a Friend*
   - */sitecore/System/Settings/Rules/Web Forms for Marketers Conditions*
   - */sitecore/System/Settings/Analytics/Reports/Subreports/Web Forms for Marketers Reports*
   - */sitecore/System/Settings/Analytics/Reports/Subreports/Web Forms for Marketers Detailed Reports*
   - */sitecore/Templates/Branches/Web Forms for Marketers*
   - */sitecore/Templates/Web Forms for Marketers*
   - All report items related to the Web Forms for Marketers module, located in the */sitecore/System/Settings/Analytics/Reports SQL Queries* folder with the icon

   - All the goal items related to the Web Forms for Marketers module, located in the */sitecore/System/Marketing Center/Goals* folder
2. In the Core database, delete the following items:
   - */sitecore/content/Applications/Content Editor/Ribbons/Chunks/Forms*
   - */sitecore/content/Applications/Content Editor/Ribbons/Contextual Ribbons/Forms*
   - */sitecore/content/Applications/Content Editor/Ribbons/Strips/Presentation/Forms*
   - */sitecore/content/Applications/Modules/Web Forms for Marketers*
   - */sitecore/content/Applications/WebEdit/Custom Experience Buttons/Edit Form*
   - */sitecore/content/Documents and settings/All users/Start menu/Programs/Web Forms for Marketers*
3. In the file system, delete the following files:
   - `\App_Config\Include\Captcha.config`
   - `\App_Config\Include\Sitecore.forms.config`
   - `\bin\ MSCaptcha.dll`
   - `\bin\ Sitecore.Forms.Core.dll`
   - `\bin\ Sitecore.Forms.Custom.dll`
   - `\bin\ System.Data.SQLite.dll`
   - `\bin_x64\ System.Data.SQLite.dll`
   - WFFM databases in the `\Data` folder
   - `\layouts\system\ VisitorIdentificationExtension.aspx`
   - `\sitecore\shell\Applications\Modules\Web Forms for Marketers`
   - `\sitecore\shell\Themes\Standard\Default\WFM`
   - `\sitecore\shell\Themes\Standard\Default\FormBuilder.css`
   - `\sitecore\shell\Themes\Standard\Default\FormDataViewer.css`
   - `\sitecore\shell\Themes\Standard\Default\MultiTreeView.css`
   - `\sitecore\shell\Themes\Standard\Default\Placeholder.css`

- `\sitecore\shell\Themes\Standard\Firefox\WFM`
- `\sitecore\shell\Themes\Standard\Firefox\FormBuilder.css`
- `\sitecore modules\Shell\Web Forms for Marketers`
- `\sitecore modules\Web\Web Forms for Marketers`

4. Publish your website to apply the changes.

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

Add a field to a web form

The Web Forms for Marketers module contains a number of default field types that you can use to build your web forms. The set of fields defines the look and purpose of a form and you can add a field to a new form or modify an existing one. You add a field to a web form through the Form Designer and you can group your fields in sections to make your form more organized and easier for users to fill out. A form can also consist only of fields that are not grouped into sections.

Each form field and section contain the following fields:

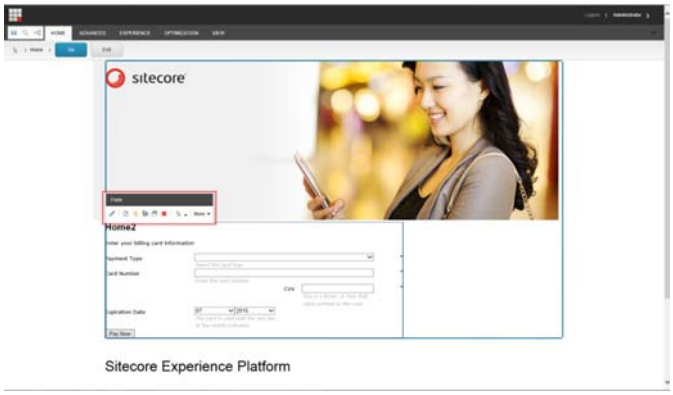
- Title: The title of the field that appears in the form
- Type: The type of field that appears; Simple, List or Complex type
- Required: Whether the visitor must fill out the relevant field
- Delete button: Deletes the field from the form

Besides adding fields, you can further personalize your web forms by configuring the submit button, configuring form verifications, as well as specifying save and success actions.

You can add a field or section to a form from the Experience Editor or from the Sitecore Desktop.

To add a field or a section to a new form:

1. To open the form that you want to add a field or section to, in the Experience Editor, open the relevant webpage and find the relevant form. Click on the form layout and the Form dialog box appears. Click the Edit Form in Form Designer icon .

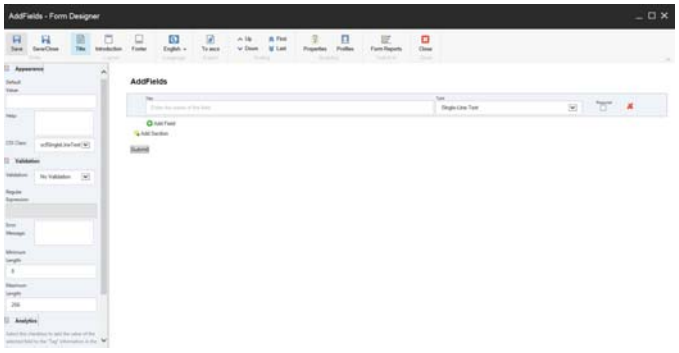


Note

From the Sitecore Desktop, click the Sitecore Start button – All Applications – Web Form for Marketers, and then click Form Designer and select the form that you want to add the field or section to.

2. In the Form Designer, click Add Field or Add Section.

A new field or a new section is added to the form. You can edit its title, type, and whether the field is required. You can see the properties of the field in the left-hand pane of the Form Designer. The list of properties depends on the type of the field selected, for example *Single-Line Text*.



3. Click Save or Save/Close to save your form.

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

Create a different language version of a web form

In the Web Forms for Marketers module, you can localize web forms by translating individual field names and other form information to other languages or dialects. The web forms created in the Web Forms for Marketers module can be fully translated into other languages. Multilanguage support is implemented for:

- Form fields and error messages
- Success pages and messages

Before you localize a web form, make sure that the required language is added to the Sitecore solution. Contact your Sitecore administrator if the language you need is not available.

This topic describes how to:

- Localize a web form field and error message
- Localize a save action, success message, or form verification message

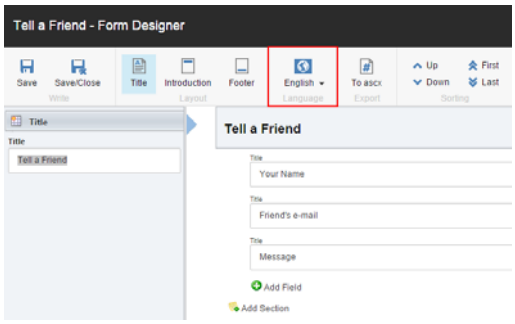
Note

You can also add a list field to a web form and localize the predefined values of the list.

Localize a web form field and error message

To localize a web form field and error message:

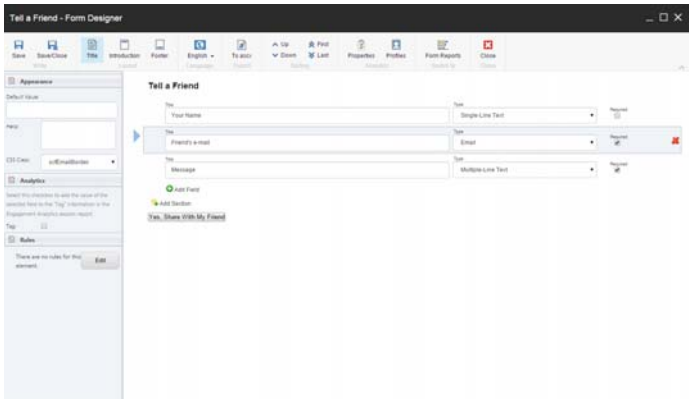
1. Open the relevant form in the Form Designer.
2. In the Language group, click the current language to display a list of the language versions that are available and select the language you want to translate the form to.



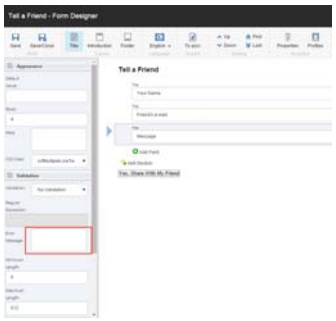
3. Enter the translated text for the fields in the form. By default, the first time the form appears the field names are blank with a help text in the original language. Unnamed fields and sections are deleted from the form.

   Note

   If you change the type of the field, it is also changed in every language version of the form. Changing the field type may result in the loss of the data stored in the field.



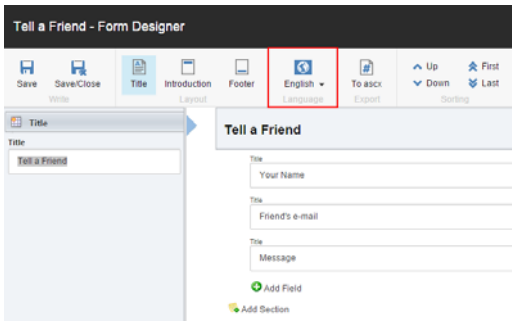
4. If a field has an error message that you want to translate, enter the translated text in the Error Message field.



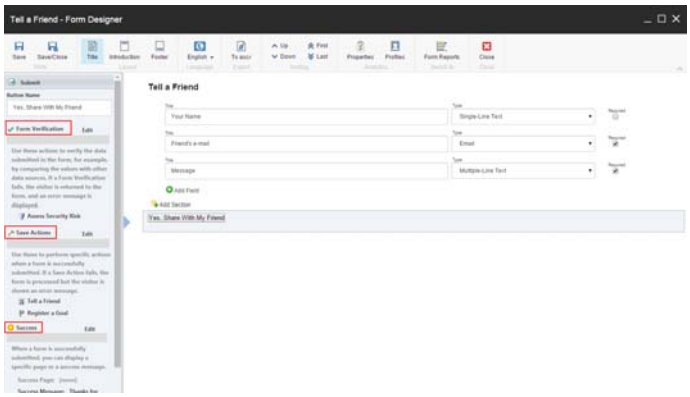
Localize a save action, success message, or form verification message

To translate save actions, error messages, success messages, or form verification error messages into any language or redirect the visitor to a page in a specific language:

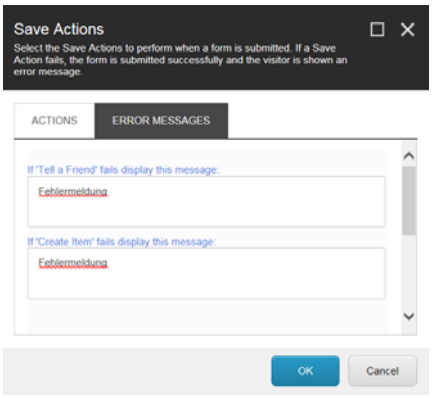
1. Open the relevant form in the Form Designer.
2. In the Language group, click the current language to display a list of the language versions that are available and select the language you want to translate the form to.



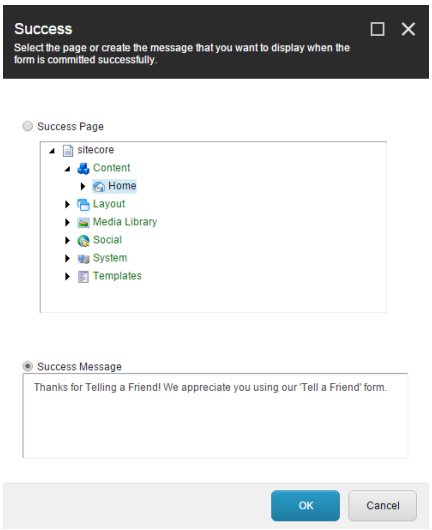
3. Click the Submit button and then click either Form Verification, Save Actions, or Success to open the relevant dialog box:



4. Either:
   - For save actions or form verification, on the Error Messages tab, enter the translated text.



   - For a success action, either specify an alternative Success Page or enter the translated text in the Success Message field.



If you do not enter any text, the default message in English is displayed to the visitor.

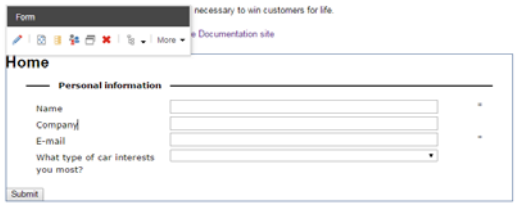
5. Click OK and save the changes to the web form.

Create a field rule in the Rule Set Editor

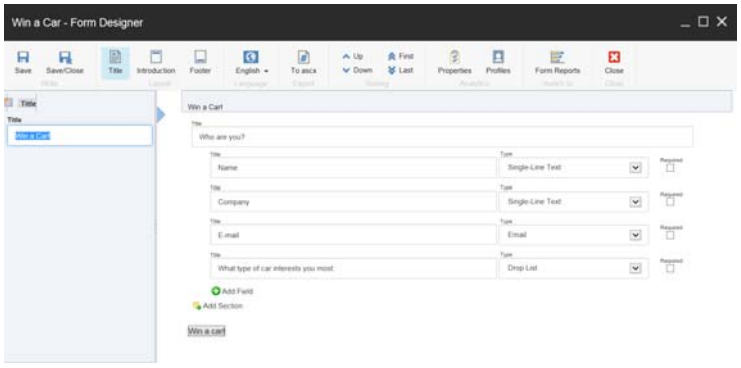
A set of one or more conditions and a corresponding action is called a rule. You can create conditions that identify the target audiences of your website and configure actions that define the behavior of a web form if the conditions are met. If you created several conditions in the form field or section, the module processes all the conditions. Conditions are processed in a top-down order.

To create a field condition and a corresponding action:

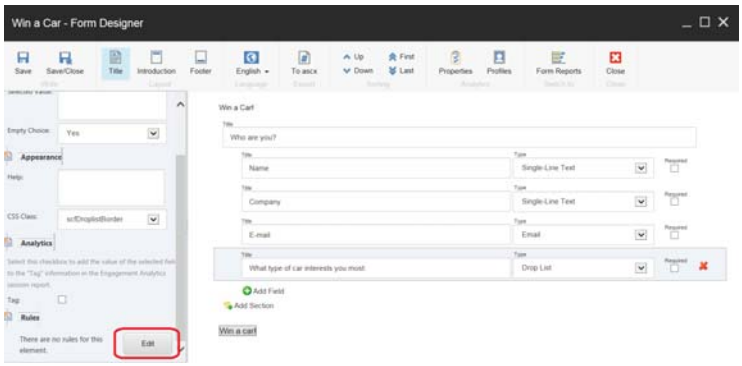
1. In the Experience Editor, click on the relevant web form and a Form dialog box appears.



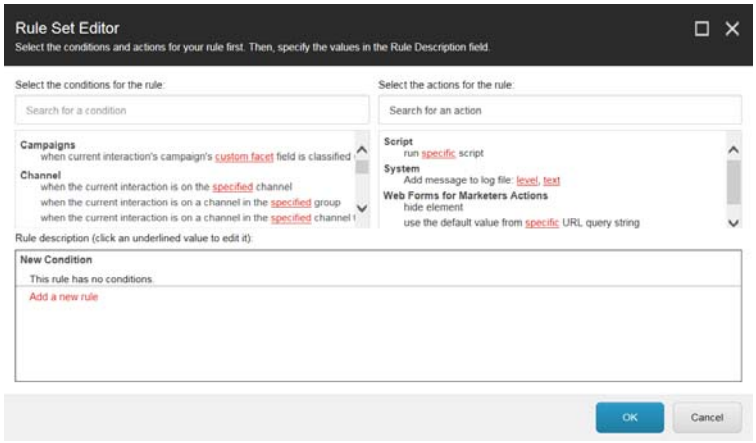
2. Click Edit form in Form Designer and the Form Designer appears.



3. In the Form Designer, select the relevant web form field and on the left pane, in the Rules section, click Edit.



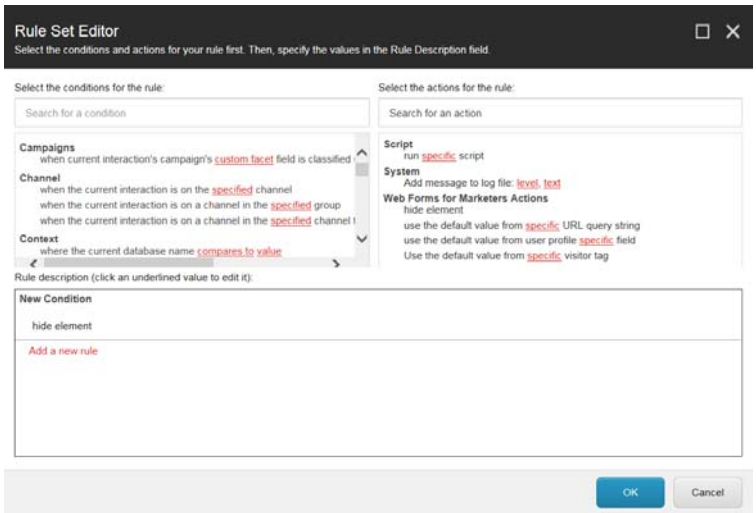
4. In the Rule Set Editor dialog box, in the Select the conditions for the rule field, select one or more conditions.
5. The selected conditions appear in the Rule description section.



6. In the Select the actions for the rule field, select the action that you want the web form to perform if the conditions are met. The available field actions are:

Field action	Description
Hide element	Hides a field or a section for the current website visitor
Use the default value from specific URL query string	Reads the value of the specific parameter in the query string and makes it the current field value
Use the default value from user profile specific field	Reads the value of the specific parameter in the user profile of the current website visitor and makes it the current field value
Use the default value from specific visitor tag	Reads the value of the specific visitor tag in the DMS database and makes it the current field value

7. The selected action appears in the Rule description section.



8. Click the underlined keywords to change or set the values. You can configure conditions using keywords, such as 'compares to', 'is greater than', 'starts with', and so on. Keywords in red indicate that you must click the keyword and enter a value. Use one of the following methods to set the keyword:
   - Click the keyword and enter the value. For example, expression, specific country, and so on.
   - Click the keyword and choose an alternative keyword. For example, compares to, where, and so on

    - Click the keyword and choose the value. For example, specific, specific template, and so on.
9. When there are no more keywords in red, click OK and save the changes.

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

Create and insert a web form

With Web Form for Marketers, you can create web forms to collect information from website visitors, for example, when they register for a newsletter or pay with a credit card. A web form records and reports all the information that visitors enter, regardless of whether they successfully submit the form, and you can view and analyze this information in the Web Forms for Marketers reports. You can easily create a new web form or copy and modify an existing one and then insert it on your web page.

In addition to web forms, Sitecore supports the creation of Model View Controller (MVC) forms and provides both native and automatic support using a standard Sitecore route. You can also combine these two approaches to run scenarios where the majority of your website is delivered as web forms while the application part of your site is delivered as MVC.

The Web Forms for Marketers module contains the following default form templates that you can edit and insert on your webpage:

- Create an Account
- Get Our Newsletter
- Pay with Credit Card
- Tell a Friend
- Demo Form
- Leave a Message

If the web form that you want to create is similar to one of these forms, you can insert a ready-made form and edit it. Alternatively, you can create a new blank web form and add your own fields and sections.

This topic outlines the following procedures:

- Create and insert a web form onto a page in the Experience Editor
- Create and insert a web form onto a page in the Content Editor
- Create a form from the Sitecore Desktop

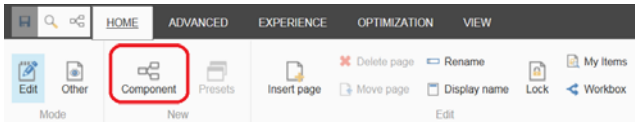
Note:

If your Sitecore administrator has restricted adding web forms to a placeholder, you cannot insert a web form into it. You can read more about placeholders in the Selecting placeholders shown in the Placeholder List topic.

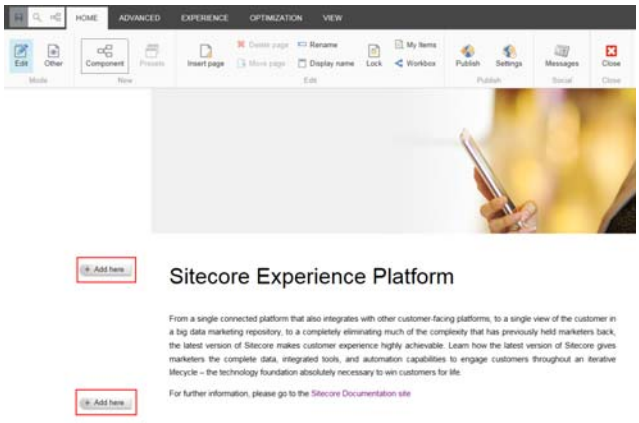
Create and insert a web form onto a page in the Experience Editor

To create a new form or modify an existing one in the Experience Editor:

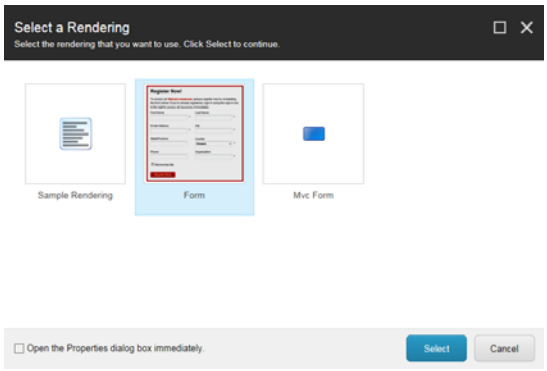
1. In the Experience Editor, on the relevant webpage, on the Home tab, in the New group, click Component.



2. The Add here buttons appear on the web page



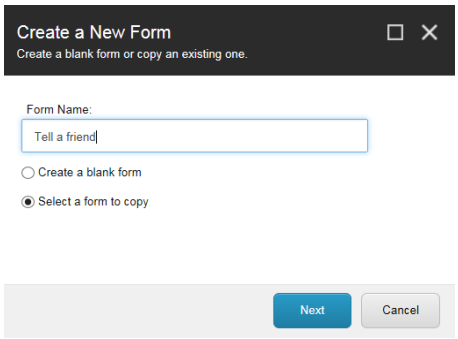
3. In the place where you want to add your web form, click Add here.
4. In the Insert a Form wizard, in the Select a Rendering dialog box, select the relevant rendering for your form, and then click Select.



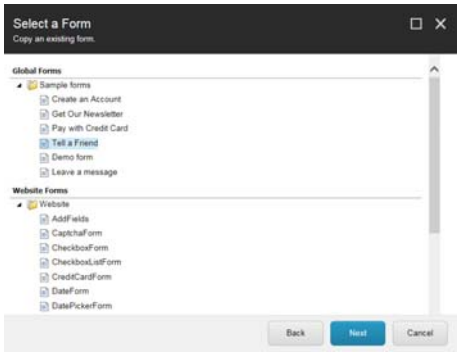
5. In the Create a New Form dialog box, enter a name for the form and either click Create a blank form or Select a form to copy and then click Next.

   Note

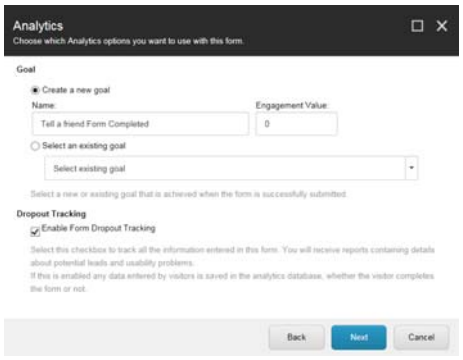
   You must give the form a unique name – you cannot have two forms with the same name.



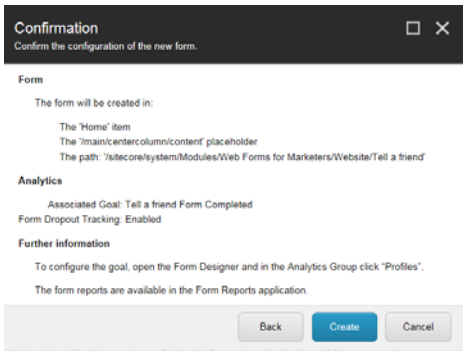
6. If you choose to copy an existing form, in the Select a Form dialog box, select the form that you want to copy, for example, *Tell a Friend,* and then click Next.



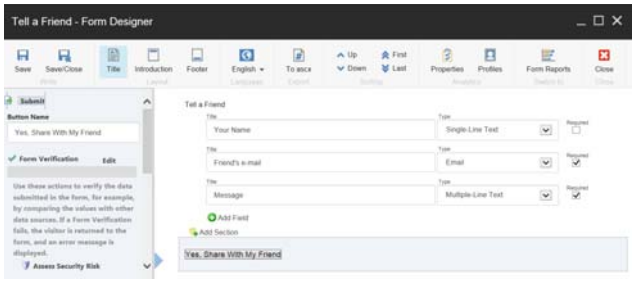
7. To assign a goal and its engagement value for your form, in the Analytics dialog box, you can either select Create a new goal and specify the engagement value for it, or you can associate an existing goal with your form by selecting Select an existing goal.



8. If you want all the information entered in the form to be tracked, regardless of whether the visitor completes the form or not, in the Dropout Tracking section, select Enable Form Dropout Tracking.
9. Click Next and in the Confirmation dialog box, verify your choices, and then click Create to insert the ready-made form.



The new form opens in the Form Designer where you add fields and sections.

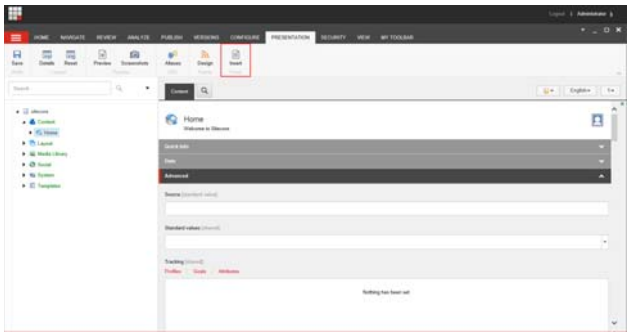


10. Click Save/Close when you are finished.

Create and insert a web form onto a page in the Content Editor

To create a new form or modify an existing one in the Content Editor:

1. In the Content Editor, on the ribbon, on the Presentation tab, click Insert.



2. In the Create a New Form dialog box, enter a name for your form and click either Create a blank form or Select a form to Copy, and then click Next.
3. If you choose to copy an existing form, in the Select a Form dialog box, select the form that you want to copy, for example, *Tell a Friend*, and then click Next.
4. In the Select a Placeholder dialog box, click the relevant placeholder where you would like your form to be located, and in the Devices field, click the drop-down arrow to specify the device, and then click Next.
5. To assign a goal and its engagement value for your form, in the Analytics dialog box, you can either select Create a new goal and specify the engagement value for it, or you can associate an existing goal with your form by selecting Select an existing goal.
6. If you want all the information entered in the form to be tracked, regardless of whether the visitor completes the form or not, in the Dropout Tracking section, select Enable Form Dropout Tracking.
7. Click Next and in the Confirmation dialog box that appears, verify your choices and then click Create.

   The new form opens in the Form Designer where you add fields and sections.

8. Click Save/Close when you are finished.

Create a form from the Sitecore Desktop

You can also create a new web form or copy or modify an existing one from the Sitecore Desktop:

1. If you are creating or modifying a form from the Sitecore Desktop, click the Sitecore Start button, All Applications, Web Form for Marketers, and click Create a New Form.
2. In the Create a New Form dialog box, enter a name for the form and then either click Create a blank form or, if you want to use an existing ready-made form to modify, click Select a form to copy and then click Next.
3. To assign a goal and its engagement value for your form, in the Analytics dialog box, you can either select Create a new goal and specify the engagement value for it, or you can associate an existing goal with your form by selecting Select an existing goal.
4. If you want all the information entered in the form to be tracked, regardless of whether the visitor completes the form or not, in the Dropout Tracking section, select Enable Form Dropout Tracking.
5. Click Next and in the Confirmation dialog box that appears, verify your choices and, to create the ready-made form, click Create.

   The new form opens in the Form Designer where you add fields and sections.

6. Click Save/Close when you are finished. You will now find this newly-created form in the template list.

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

Edit, move or delete a web form

You use web forms to collect information about your website visitors who supply information when they fill in a form. You can then retrieve this information from the Web Forms for Marketers reports. Once forms have been placed on your website, you can edit, move or delete them directly from the Experience Editor.

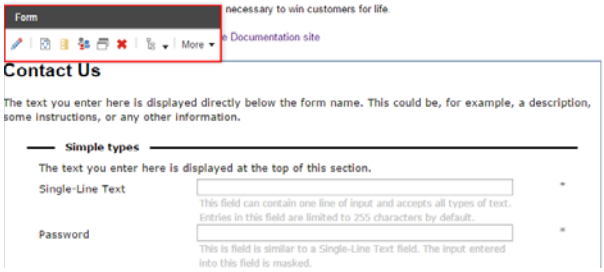
In the Experience Editor, you can:

- Edit a web form
- Move a web form
- Delete a web form

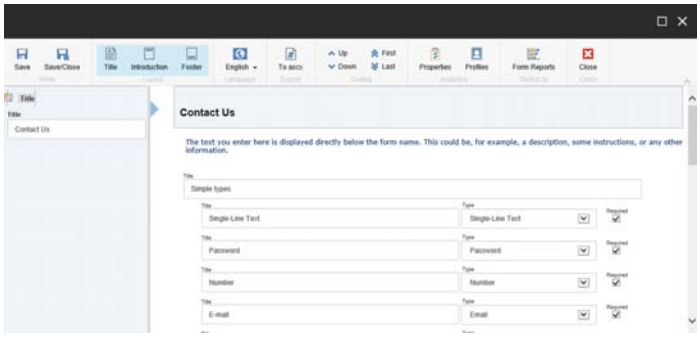
Edit a web form

To edit an existing web form in the Experience Editor:

1. In the Experience Editor, open the relevant page and click on the form that you want to edit. The Form dialog box will appear on the form.



2. Click Edit and when the Form Designer opens, edit your web form.



Note

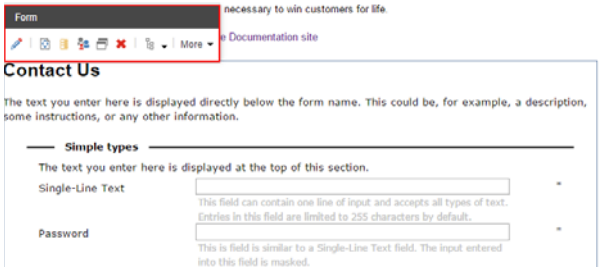
You can also open the Form Designer by opening the Sitecore Desktop and clicking on the Sitecore Start button, All Applications, Web Forms for Marketers, and click Create a New Form.

3. To save your changes, click Save or Save/Close.

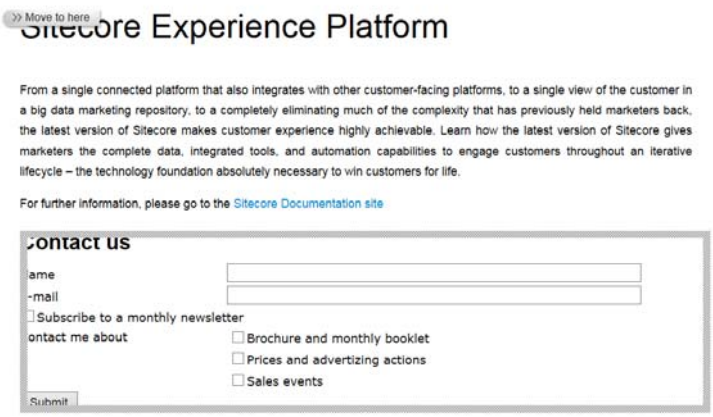
Move a web form

To move an existing form to another place:

1. In the Experience Editor, on the relevant page, click the form that you want to move. The Form dialog box will appear on the form



2. Click Move Component. The Move to here buttons appear in all the places on the page where you can move the web form to.

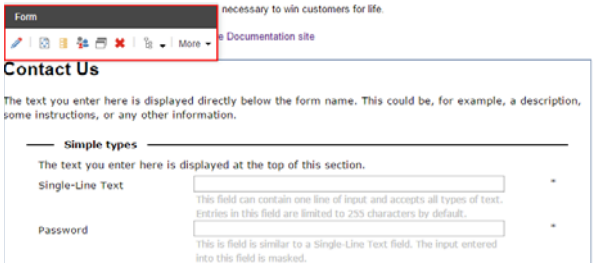


3. Click Move to here where you want to move the web form to.
4. To save the new position of the form, click Save.

Delete a web form

To delete an existing form:

1. In the Experience Editor, on the relevant page, click the form on the page that you want to delete:



2. Click Delete.
3. To save the changes, in the Experience Editor, click Save.

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

Insert a web form directly on a web page

After an MVC (ASP.NET MVC) or non-MVC (ASP.NET Web Forms) form is created in the Form Designer, users can insert it onto a webpage using the Experience Editor or Content Editor.

However, developers and administrators can insert an existing web form directly onto a web page in the following ways:

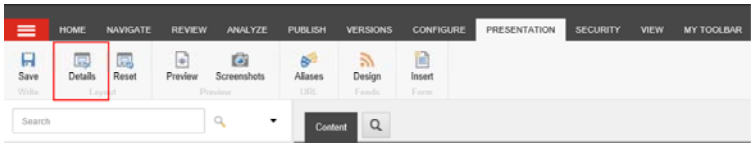
- Insert an existing web form onto a page in the Content Editor
- Insert an existing web form onto a page through a web control
- Insert an existing web form onto a page using the code-behind class

Insert an existing form onto a page in the Content Editor

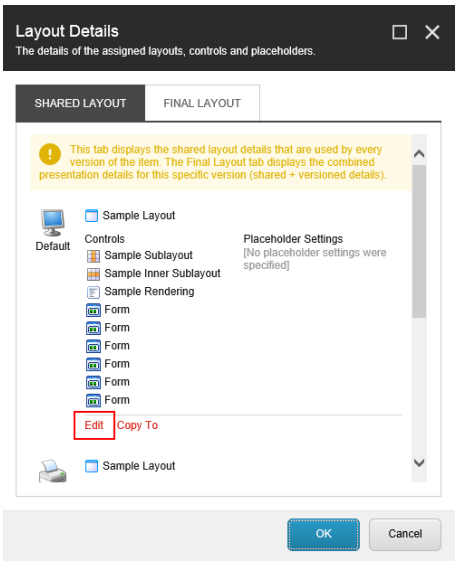
A web form is a Sitecore rendering, so you can insert it in the Content Editor.

To insert an existing form onto a page in the Content Editor:

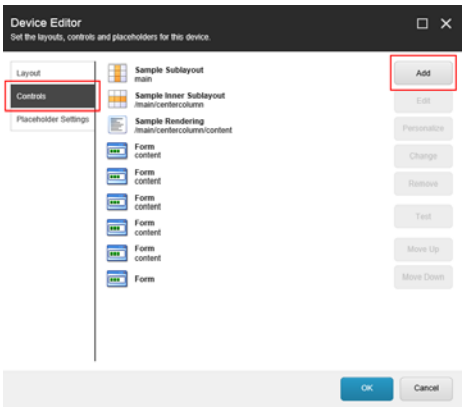
1. In the Content Editor, click the item that you want to add a web form to.
2. On the ribbon, in the Presentation tab, in the Layout group, click Details.



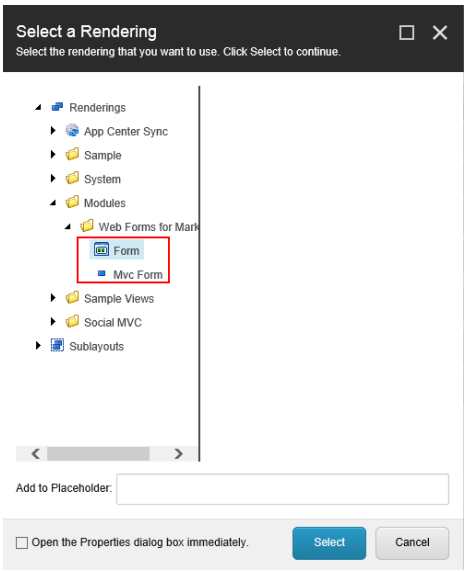
3. In the Layout Details dialog box, on the Shared Layout tab, select the relevant device, and then click Edit.



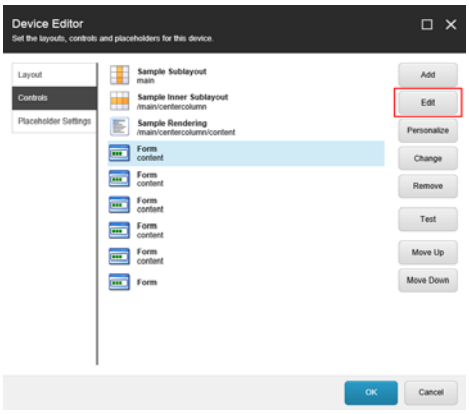
4. In the Device Editor dialog, in the left pane, click Controls and then click Add.



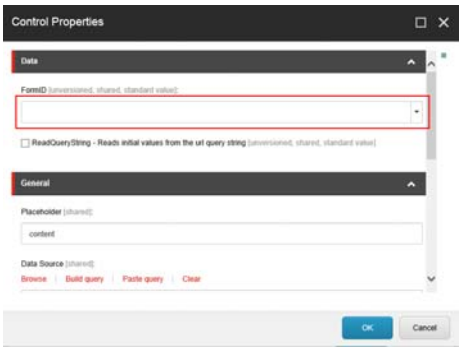
5. In the Select a Rendering dialog, navigate to *Renderings/Modules/Web Forms for Marketers* and select either Form or MVC Form, then in the Add to Placeholder section, enter the [placeholder name] and click Select.

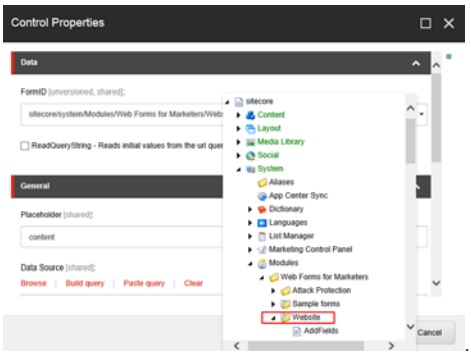


6. In the Device Editor dialog, click the new control, and then click Edit.



7. In the Control Properties dialog, in the Data section, in the FormID field, click the drop-down arrow and navigate to the *System/Modules/Web Forms for Marketers/Website* folder, select the relevant web form, and then click OK.





8. In the Device Editor dialog and Layout Details dialog, click OK, and then click Save.

   The selected form is now displayed on the website in the selected placeholder.

Insert an existing form onto a page as a web control

You can add a web form to a layout statically by using .aspx, .ascx, or .cshtml layouts.

To insert an existing form onto a page as a web control:

- For an .aspx or .ascx layout:
  1. Open an .aspx or .ascx file and add the following code to the namespace:

```
<%@ Register TagPrefix="wffm" Namespace="Sitecore.Form.Core.Renderings" Assembly="Sitecore.Forms.Core" %>
```

  1. Add the `FormRenderer` tag:

```
<wffm:FormRender  FormID="<id of the form item>" runat="server"/>
```

- For a .cshtml layout:
  - Open a .cshtml file and add the following code:

```
@{ Sitecore.Mvc.Presentation.RenderingContext.Current.Rendering.Parameters["FormId"] = "<id of the form item>"; }
@Html.Sitecore().Controller("Sitecore.Forms.Mvc.Controllers.FormController, Sitecore.Forms.Mvc")
```

  - Alternatively, you can also use the following code:

```
@Html.Sitecore().Rendering("{F2CCA16D-7524-4E99-8EE0-78FF6394A3B3}", new { Datasource = "<id of the form item>" , UniqueId = "<un
```

Insert an existing form onto a page using the code-behind class

You can add web forms depending on different conditions using the code-behind class.

To insert an existing form onto a page using the code-behind class:

```
FormRender fr = new FormRender();
fr.FormID = "5D9E85F3-5E03-49A7-A136-93269DEA22A7";//form item id
Sitecore.Context.Page.GetPlaceholder("main").Controls.Add(fr);
```

This sample code inserts a web form with the specified `ID` into the `Main` placeholder.

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

Setting up a web form

The Web Form for Marketers module enables marketers, content authors, and others to create and insert new forms, customize default ones, add fields, as well as configure the submit button in a simple and straightforward manner. The module also records and reports the information provided by visitors in forms regardless of whether the visitor fills in the form successfully or not.

To set up a web form, you need to follow these steps:

- Create a web form
- Add a field to a web form
- Configure the submit button

Create and insert a web form

You can create new forms from scratch, modify existing form templates and then insert both web forms and MVC forms in a simple manner onto a page. This can be done in the Experience Editor, Content Editor or from the Sitecore Desktop.

Add a field to a web form

Adding fields and sections to a web form is an integral part of making web forms work successfully. The set of fields defines the look and purpose of a form and you can edit its title, type and whether the field is required or not.

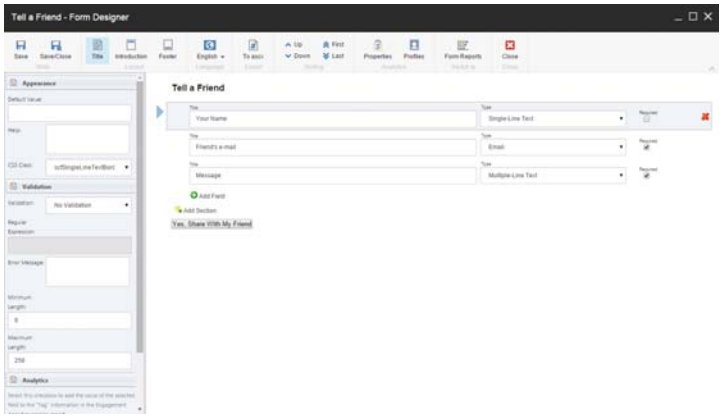
Configure the submit button

You can also configure the submit button, enabling you to set up the appropriate form verification as well as specify save and success actions.

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

The Form Designer

You use the Form Designer to design and edit web forms. On the right, you can create form sections and fields and in the pane on the left, you can configure properties for all the elements of your web form.



The Form Designer toolbar groups

Element	Description
Write group	Click Save to save changes or Save/Close to save changes and close the form.
Layout group	Click Title, Intro and Footer to create a form title, some introduction text, and a form footer (the text displayed under the Submit button on your form).
Language group	Click Language to switch the current language of the form.

	Note In the Content Editor, you can create a different language version of a web form.
Export group	Click To ascx to convert your form into a standard ASCX control.
Sorting group	Click one of the sorting buttons to sort a selected field or section.
Analytics group	Click Properties and Profiles to set up the analytics options.
Switch to group	Click Form Reports to switch to the Form Reports application.

Web form field properties

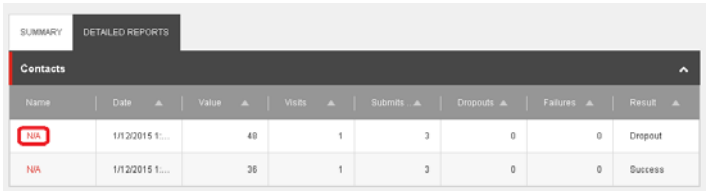
Element	Description
Appearance	Contains the parameters that affect how the field is displayed on the form: default value, help text and CSS class.
Validation	Contains the parameters that define the range of acceptable values for the field: validation type, the appropriate expression, as well as the minimum and maximum number of symbols.
Analytics	Contains the Tag parameter. If you select the Tag check box for a field, the information in this field is displayed in the Session Reports of Sitecore Engagement Analytics (SEA).
Rules	Contains rules that define the behavior of the selected web form field or section.

Web form elements

Element	Description
Title (the form title)	The name of the form displayed at the top of the form, specified in the Layout group.
Introduction	The text displayed after the title. This may include any useful information that website visitors should know before they fill out your form
Fields	Contains the title and type of fields and whether they are required.
Sections	Contain the headings that fields are grouped under. This is optional.
Submit button	Triggers the actions associated with the form. You can specify the name of this button in the pane on the left and configure the form verification, save actions, and success message.

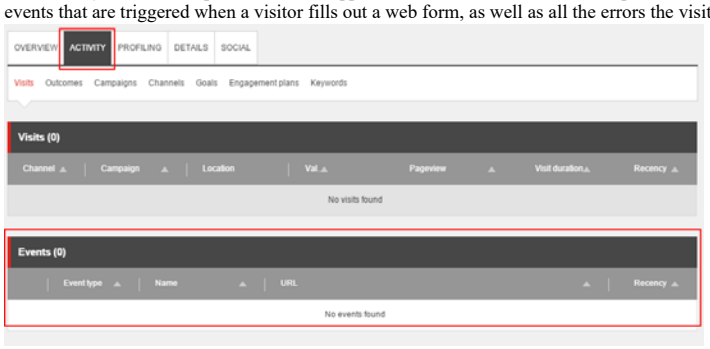
Form reports events

You can view web form events by clicking the required report in Form Reports, on the Detailed Reports tab. You can access the Form Reports from the Sitecore Desktop.



The web forms events are displayed in the Experience Profile application, which you can access from the Sitecore Launchpad.

The Activity tab in the Experience Profile application lists the activities that are performed when a site visitor is on the website. The Events section displays all the events that are triggered when a visitor fills out a web form, as well as all the errors the visitor encountered when interacting with web forms.



In the Activity section, you can find both field events and form events.

Several field events can occur in the same field during the same attempt to submit a web form, as users change the information in the field.

The following events are listed in the form reports:

Event	Description
Field Completed	Triggered when a field in a web form contains a value and is tabbed or clicked out of. You can implement this using AJAX.
Field Not Completed	Triggered when a validation of a required field fails because the field has not been filled in by the visitor.
Field Out of Boundary	Triggered when a validation of a field fails because the value entered falls outside the boundaries defined for the field. This event is used for the Min and Max Length of the Text and Password fields, as well as for Date and Number fields.
Form Check Action Error	Triggered when a *Check Action* fails. When this event occurs, there is a *Check Action* error and the visitor is returned to the form. No other *Check Actions* or save actions are initiated.
Form Save Action Failure	This is a failure action, and is triggered when the *IsFailure* property is enabled.
Form Submit	Triggered when a visitor clicks the *Submit* button or presses the Enter key. This indicates an attempt to submit the form.
Invalid Field Syntax	Triggered when a validation of a field fails because the data entered in the field fails a particular syntax check. The event is used for the following field validations: *Regular Expressions in Text*, *Regular Expressions in Password types*, and *Email fields*.
Submit Success	Triggered when a *Submit* action does not return an error. The event is registered together with the *Form Conversion* event, and is primarily used to facilitate the SQL statements required for the Form Dropout, Form Usability, and Form Failures reports.
Form Begin	Triggered when a visitor starts filling out or submitting a web form. Note The *Form Begin* event is critical to reporting. You should not manually trigger it because this can affect form dropout reporting.
Form Dropout	Triggered when the user session has expired and the visitor did not successfully submit the web form. Note

The *Form Dropout* event is critical to the reporting. You should not manually trigger it because this can affect form dropout reporting.

Open web form reports

Information about how site visitors interact with web forms is stored in reports that are part of the Web Forms for Marketers module. Form reports are part of the integrated analytics of Sitecore and provide detailed information about a specific web form. These statistics let you analyze the most popular entered values, and the information collected can also be easily exported to either a Microsoft Excel or XML file.

The web form reports contain information about the following:

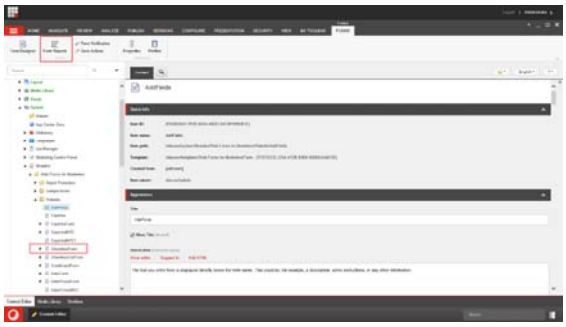
- Summary of all form responses
- Detailed reports of form responses
- Additional details

To open the form reports:

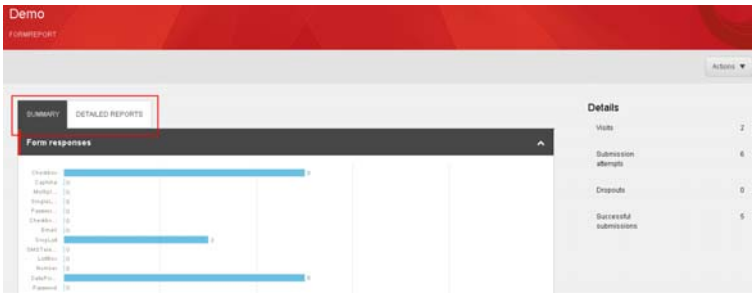
1. Log in to the Sitecore Desktop.
2. Click the Sitecore Start button, All Applications, Web Form for Marketers, and then click Form Reports.
3. Select the web form that you want to view reports for.

Alternatively, in the Content Editor:

1. In the content tree, navigate to `/sitecore/System/Modules/Web Forms for Marketers/Website` item and select the relevant form or field in a form. The Forms tab will appear in the ribbon.
2. On the Forms tab, click Form Reports.



In the report, click the tabs to see either a summary or a detailed view of the form responses. You can see web form reports on two tabs in the Form Reports window:



The web form reports

Web form reports provide information about the contacts that interact with your web forms. The reports have a number of fields providing detailed information about a specific web form.

When you open Form Reports, the Summary and Detailed tabs display the reports and the Actions button enables you to export the data to Microsoft Excel or XML format.

Report	Description
Summary	Displays a bar chart indicating the number of submitted forms. The fields of the web form are displayed on the left, and the corresponding bar indicates the number of times that a field has been filled in.
Detailed Reports	Lists the contacts that have interacted with the web form. Each row contains the following information:

- Name – the business name, based on a contact's public IP address.

  Note

  The name row is displayed only if you have Sitecore IP Geolocation service enabled.

- Date – the date that the contact last submitted a report.
- Value – the engagement value that you have assigned to the web form and that the visitor has earned by submitting it.
- Visits – the total number of visits to the page containing the web form that the specified contact interacted with.
- Submission attempts – the number of times that the submit button was clicked on the web form by the specified contact in the report.
- Dropouts – the total number of times that the web form was not submitted by a specified contact, even though some of the fields were filled in.
- Failures – the total number of failed save actions that were registered on the web form submission. A web form can be submitted even when save actions have failed.
- Result – a successful or failed submission by the specified contact.

Details	Displays summaries about the following web form submission details. • Visits – the total number of visitors who have interacted with the web form in the page. • Submission attempts – the total number of times that visitors clicked the submit button. • Dropouts – the total number of times that visitors filled in form fields but did not submit the form. • Successful submissions – the total number of times that visitors successfully submitted the web form and data was collected.

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

Walkthrough: Configure engagement analytics in a web form

The Web Forms for Marketers module is closely integrated with Sitecore Analytics, which provides functionality for recording and reporting visitor information.

When you fill in and successfully submit a form, the Analytics database records a goal conversion. If a Register Goal save action fails, the Submit Success Event is still registered, but the Goal Conversion event is not registered.

You can also attach and configure profile values to the successful submission of forms.

To configure engagement analytics to work in web forms, you need to do the following:

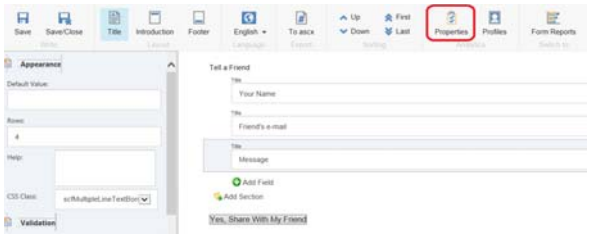
- Associate a goal with a form
- Configure profile values for the associated goal
- Associate a campaign with the web form submission

Associate a goal with a form

When you create a web form, by default a *Form_Name Form Completed* goal is created and associated with the web form. When you associate a goal with a form, you can also enable form dropout tracking. For example, you can use a *Form Dropout* reports, to see a list of potential sales leads that have shown interest in a product or service but have not successfully registered their details.

To associate a goal with a web form:

1. In the Form Designer, in the Analytics group, click Properties.



2. In the Analytics dialog box, in the Goal field, click the drop-down list and select a goal that you want to associate with the form.

Note

You can only associate one goal and/or one campaign with each web form.

3. To track visitors who attempt to submit forms, select the Enable Form Dropout Tracking checkbox. By selecting this checkbox, you can track all the information entered in the form, regardless of whether the visitor submits the form.

   Note

   The values entered in the Password, Password confirmation, and Credit card fields are not tracked.

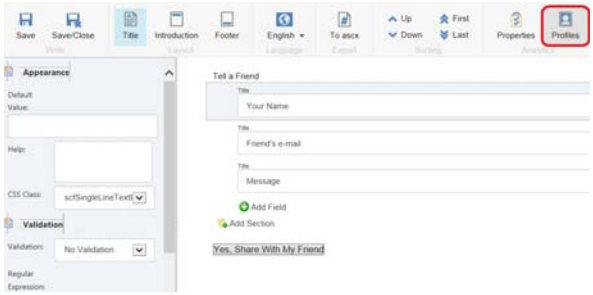
4. In the Confirmation dialog box, make sure the configuration of the form is correct and click Close.

Configure profile values for an associated goal

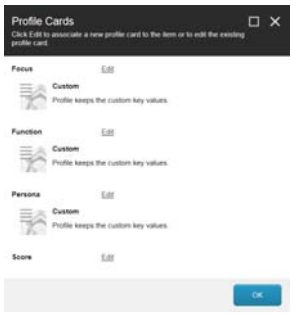
Profile cards can be used to identify and segment contacts as well as implement personalization rules. They contain saved profile keys and profile values, which track how visitors interact with the website. You can configure these profile values for goals that are associated with a web form.

To configure profile values for a goal associated with a form:

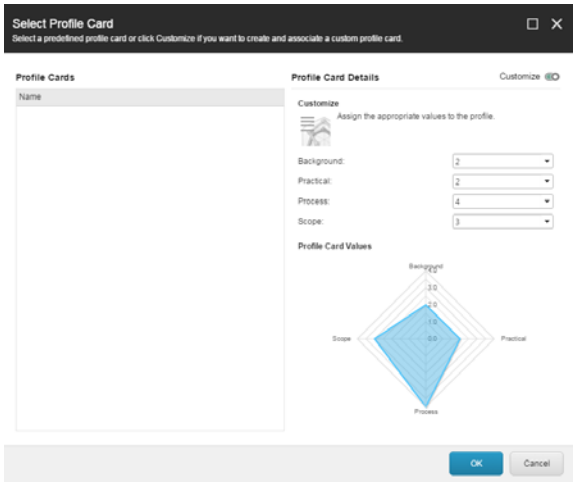
1. In the Form Designer, in the Analytics group, click Profiles.



2. In the Profile Cards dialog box, click Edit to associate a new profile card with the respective item.



3. In the Select Profile Card dialog box, set the content profile values. The default values range from 0 to 10. However, these values can be modified.



4. To save the changes, click OK.

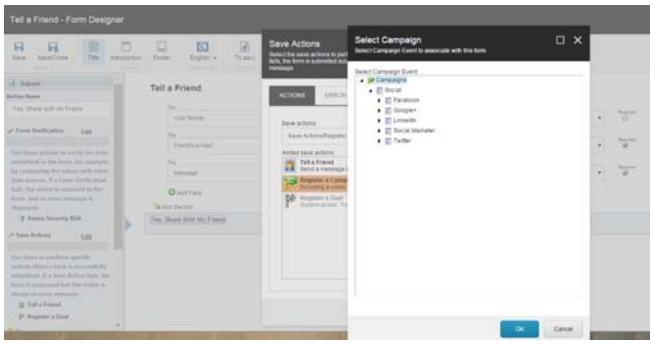
Associate a campaign with a form submission

You can associate a campaign with a web form, to make a visitor a member of a campaign when they successfully submit the form.

To trigger a campaign event upon form submission, you must add the appropriate Register Campaign action to the form's Save Actions list.

To associate a campaign with the form:

1. In the Form Designer, click the Submit button and in the left pane click Save Actions.
2. In the Save Actions dialog box, in the Save Actions field, select the *Register a Campaign* action from the drop-down list and then click Add.
3. In the Added Save Actions field, select Register a Campaign and then click Edit.



4. In the Select Campaign dialog box, in the Select Campaign Event field, select the campaign that you want to make the visitor a member of and click OK.
5. To save your changes, in the Save Actions dialog box, click OK.

Add and customize a reCAPTCHA field on MVC layouts

You can protect your website from robot attacks, spam, and abuse by implementing a CAPTCHA field for ASP.NET (non-MVC) layouts, or a reCAPTCHA field for MVC form renderings. Both CAPTCHA and reCAPTCHA fields require the site visitor to enter a combination of displayed symbols.

Note

The redirect functionality in Robot Attack Protection does not apply in the reCAPTCHA field. Instead, CAPTCHA will be shown on the same page.

To set up and configure reCAPTCHA, perform the following steps:

1. In your web browser, ensure that you enable JavaScript.
2. Register your website at: https://www.google.com/recaptcha/admin#list

   Upon registration, you receive a Secret key and a Site key to add reCAPTCHA to your site.

3. Add the following settings to the `Sitecore.MvcForms.config` file in the `\App_Config\Include\` folder:

```
<!-- Recaptcha private/secret key -->
      <setting name="WFM.RecaptchaSecretKey" value="" />
 <!-- Recaptcha public/site key -->
      <setting name="WFM.RecaptchaSiteKey" value="" />
</settings>
```

4. Insert the Secret and Site keys in the appropriate setting value.

   Note

   You can specify the keys for reCAPTCHA version 1.0. In this case, add the keys to the `web.config` file into the `<appSettings>` node:

```
<add key="RecaptchaPublicKey" value="[site key value goes here]" />
<add key="RecaptchaPrivateKey" value="[secret key value goes here]" />
```

5. To change the error message for the reCAPTCHA field, navigate to and select the *Captcha* folder *(/sitecore/system/Modules/Web Forms for Marketers/Settings/Field Types/Complex/Captcha)* and in the right pane, in the Data tab, in the Localized Parameters field, insert a new error message between the tags.

   Note

   For further details visit the reCAPTCHA Developer's Guide website: https://developers.google.com/recaptcha/intro

6. To customize the appearance and type of the reCAPTCHA field, in the right pane, in the Parameters field:
   - Change the color theme from light to dark – insert `<Theme>dark</Theme>`.
   - Change how the reCAPTCHA field is displayed from image to audio – insert `<CaptchaType>audio</CaptchaType>`.

Note

Web Forms for Marketers does not support parametrization of the *Data-callback* setting through the Content Editor, however you can customize the *Recaptchafield.cshtml* file.

Configure a list field on a web form

In a web form, you can specify options that you want displayed in a list field.

A list field lets users select one or more options from a predefined group of values. List fields can have both a value and text. The value is stored in the database and used for statistics.

The text is what visitors see on the form, and this can be different from the value, for example, you can enter a user-friendly text in the Text field.

You can also use the Text field to add different language versions of a list item to your web form, enabling you to localize existing list field items.

This topic describes how to:

- Add a list field to a web form by entering list items manually
- Add a list field to a web form by selecting Sitecore items
- Add a different language version of a Sitecore list item

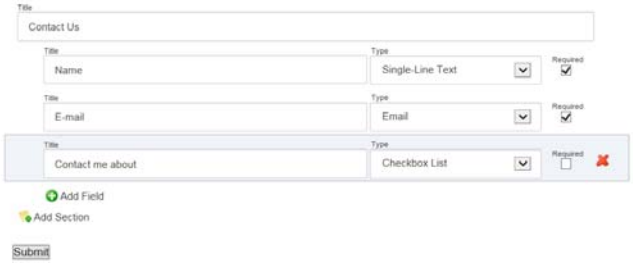
Note

You can also specify the items to display in a list field by selecting Sitecore items with the help of XPath queries, Sitecore queries, or fast queries.

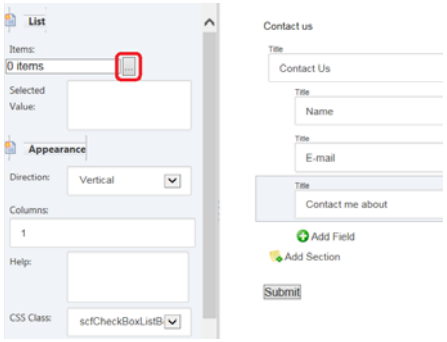
Add a list field to a web form by entering list items manually

To manually enter the values that you want to display in a check box list:

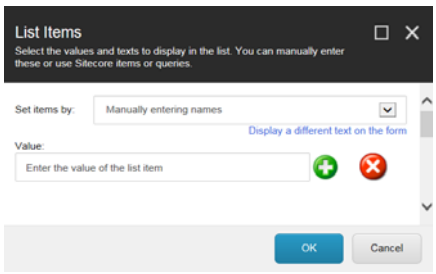
1. In the Form Designer, in the Type field for the relevant field, select the type of list field that you want to add, for example *Checkbox List*.



2. In the left pane, in the List section, click the Browse button by the Items field.



3. In the List Items dialog, in the Set items by field, click the drop-down arrow and select *Manually entering names.*

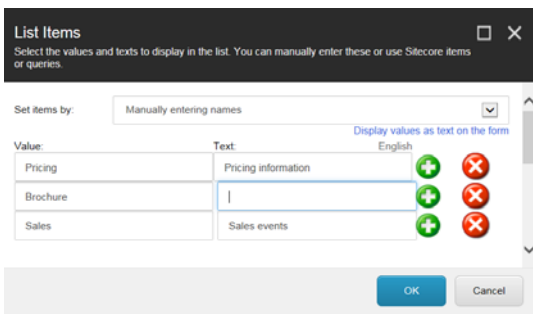


4. To display a text field where you can enter a customized, user-friendly text, click Display a different text on the form.

   Note

   List fields can have both a value and text. The value is stored in the database and used for statistics. This can also be what visitors see on the form.

5. In the Value field, enter the value that is stored in the database. To show a different text on the form, in the Text field, enter the text that you want to display.



   Note

You must always enter something in the Value field. If you leave the Text field blank (or if you click Display values as text on the form), the text entered in the Value field is displayed on the form. If there are two or more identical values in the Value field, only the first value is saved.

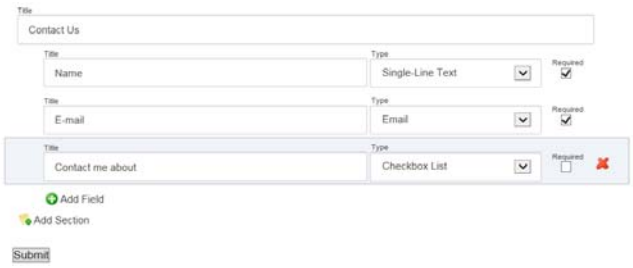
6. To add a new line, click .When you have finished entering the text to display on the form, click OK.
7. Click Save or Save/Close to save the changes.

The form looks something like this:

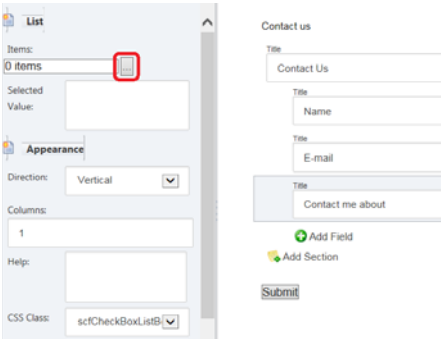
Add a list field to a web form by selecting Sitecore items

To select Sitecore options to display in a list field:

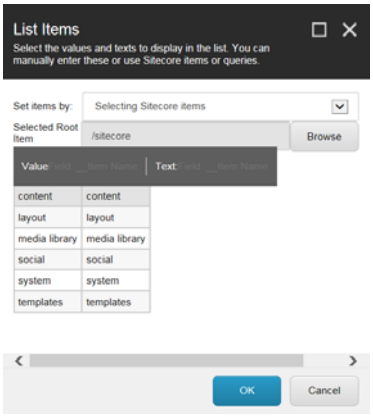
1. In the Form Designer, in the Type field for the relevant field, select the type of list field that you want to add, for example *Checkbox List*:



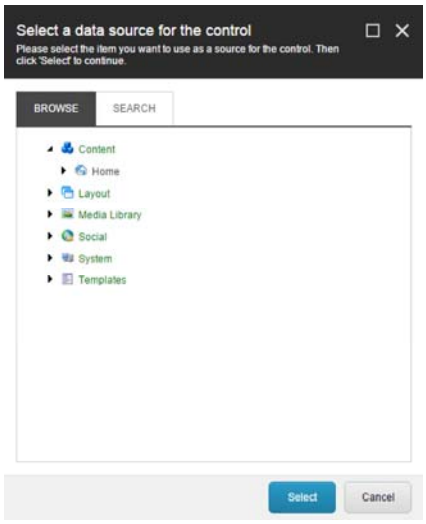
2. In the left pane, in the List section, click the Browse button by the Items field.



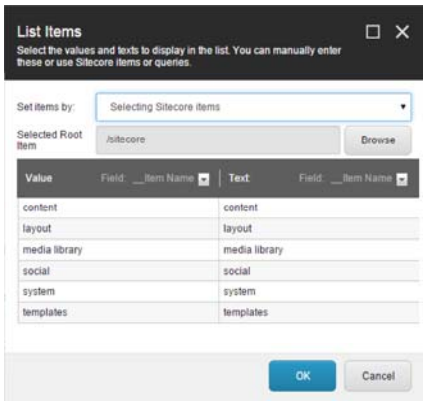
3. In the List Items dialog box, in the Set items by field, click the drop-down arrow, and select *Selecting Sitecore Items*.



4. In the Selected Root Item field, click Browse.
5. In the Select a data source for the control dialog box, navigate to the item on your website that contains the subitems corresponding to the field values that you want to display. For example, the *Customers information* item contains subitems corresponding to subscription types. Select the relevant item, and then click Select.



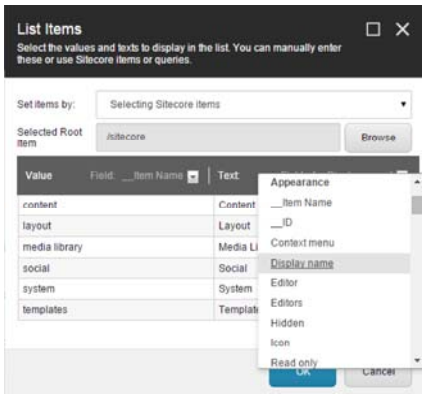
In the List Items dialog box, the subitems are now listed. The values in the Value column are stored in the database while the values in the Text column are displayed on the form.



Note

By default, Sitecore displays the Item Name field in the Value column, but you can select any of the other fields as the value. You can also display a different field in the Text column. This enables you to display an alternative, more user-friendly label on the form, for example, by selecting the Display name field.

6. To select another item field, click in the Text or Value column and select the relevant field:



7. Click OK.
8. In the Form Designer, click Save or Save/Close to save the form changes.

This is how the list field looks on your site:

Add a different language version of a Sitecore list item

You can localize the items in the list to a different language version, but only if these items have the relevant language version.

1. In the Form Designer, open the web form that you want to translate.

2. In the ribbon, in the Language group, click the relevant language.
3. Select the list field and in the left pane, click the Browse button by the Items field.
4. In the List Items dialog box, in the Text column, replace the text in the fields with the relevant translations.

   Note

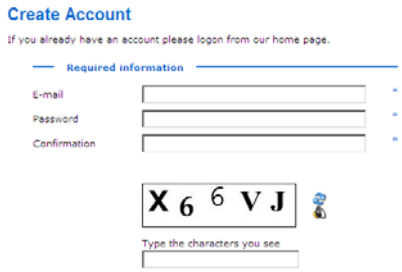
   To add a different language version of a list item manually, in the List Items dialog box, click Display a different text on the form. In the Text field that appears for each value, enter the relevant translations.

   The value displayed in the Text column is shown on the form. If the field specified in the Text column is blank, the translated version will not contain anything in this field.

Display a CAPTCHA field

The Web Forms for Marketers module gives you the tools to protect your website from robot attacks. The CAPTCHA field can help you to distinguish between real users and robots and therefore protect your website and web services from abuse by programs masquerading as real users. A CAPTCHA field requires the user to enter some symbols displayed in a field.

Here is an example of how the CAPTCHA field might look on a website:



You can choose to display the CAPTCHA field every time for every form, or you can display the field in the following cases:

- *The visitor is a robot*. The system identifies the current visitor as a robot using a special algorithm.
- *A suspicious visitor is detected*. A visitor submits the form several times in a short period of time.
- *Suspicious form activity is detected*. The form is submitted several times in a short period of time by one or more users.

  If your Sitecore solution is running on several servers then the number of form submits per server is taken into account.

You can select the condition that must be fulfilled before the CAPTCHA field is displayed on the form.

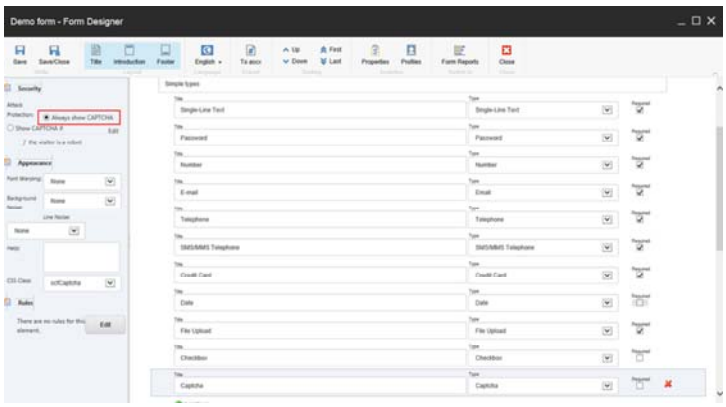
This topic outlines how to:

- Always display a CAPTCHA field.
- Display a CAPTCHA field when a suspicious visitor is detected.
- Display a CAPTCHA field when suspicious form activity is detected.
- Configure a warning email notification.

Always display a CAPTCHA field

You can configure the CAPTCHA field so that it is always displayed on the form:

1. In the Form Designer, select the CAPTCHA field.
2. In the Security section, select Always show CAPTCHA:



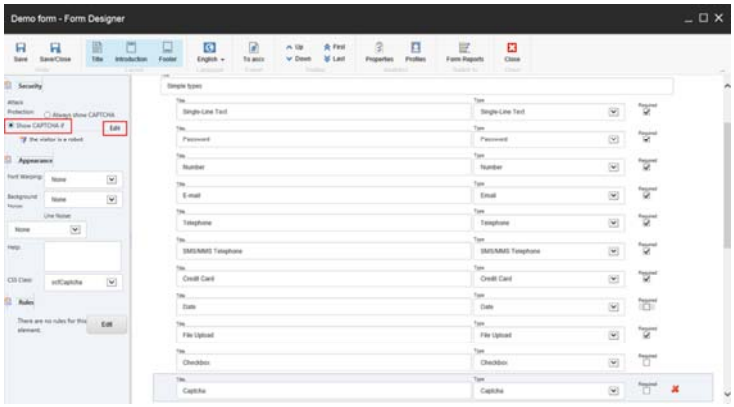
3. Click Save or Save/Close to save the form changes.

Display a CAPTCHA field when a suspicious visitor is detected

You can configure the CAPTCHA field so that it is only displayed if a robot attacks your website. A suspicious visitor is a visitor who submits a form several times in a short period of time. Usually such behavior is particular to robots.

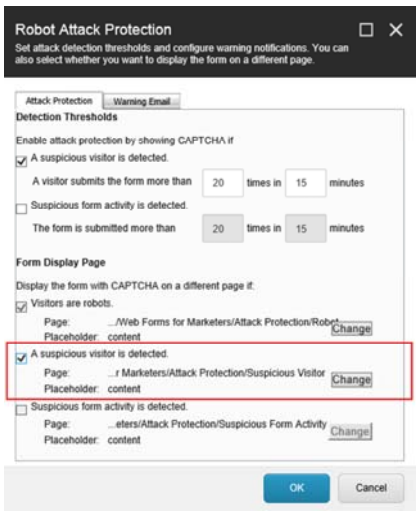
To display the CAPTCHA field if a suspicious visitor is detected:

1. In the Form Designer, select the CAPTCHA field.
2. In the Security section, select Show CAPTCHA if and click Edit.

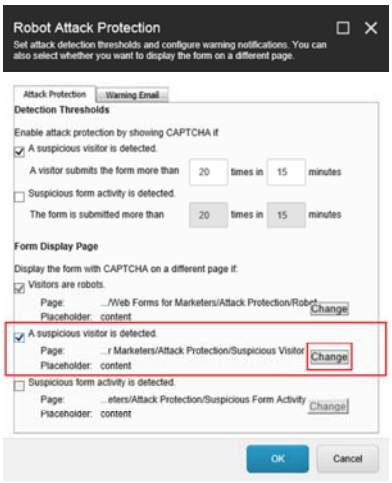


3. In the Robot Attack Protection dialog box, in the Detection Thresholds section, select a suspicious visitor is detected and enter the maximum number of times a visitor can submit the form and the time period.

4. To display the form with the CAPTCHA field on a separate page, in the Form Display Page section, select a suspicious visitor is detected.

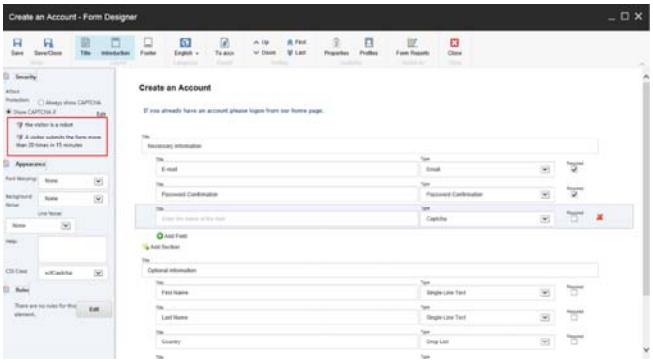


5. To display the form with the CAPTCHA field on your custom page:
   - Click Change.



- In the Select Form Display Page dialog box, browse to the page you want to display the form with the CAPTCHA field on.
- Select a placeholder for where to put the form. Contact your Sitecore administrator to find out if this placeholder is enabled using the Select Placeholders wizard.

6. Click OK to close the Robot Attack Protection dialog box.
7. Click Save or Save/Close to save the form changes.
8. When you have selected a suspicious visitor is detected, this information is displayed in the Security section, along with the time limits that you configured:



Note

When you select a suspicious visitor is detected, the visitor is a robot condition is automatically enabled.

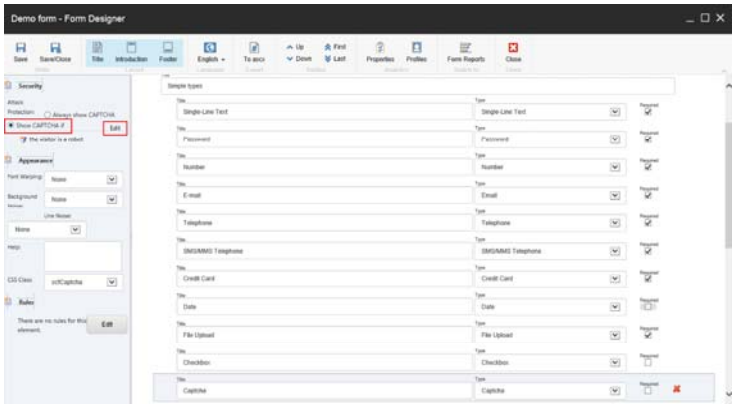
If a visitor submits the form more than the specified limit, the form with the CAPTCHA field is displayed.

Display a CAPTCHA field when suspicious form activity is detected

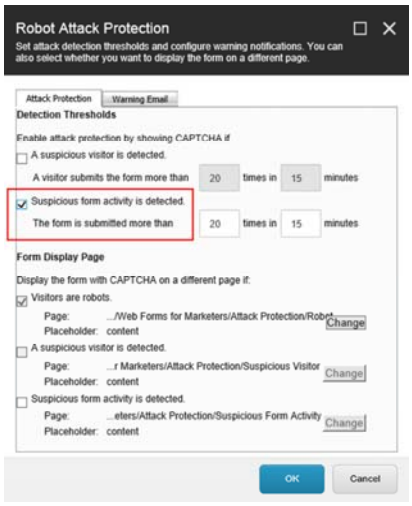
Suspicious form activity means that a form is submitted several times by several users in a short period of time. Usually such behavior is particular to robot attacks. If your Sitecore solution is running on several servers then the number of form submissions per server is taken into account.

To display the CAPTCHA field if suspicious form activity is detected:

1. In the Form Designer, select the CAPTCHA field.
2. In the Security section, select Show CAPTCHA if and click Edit.



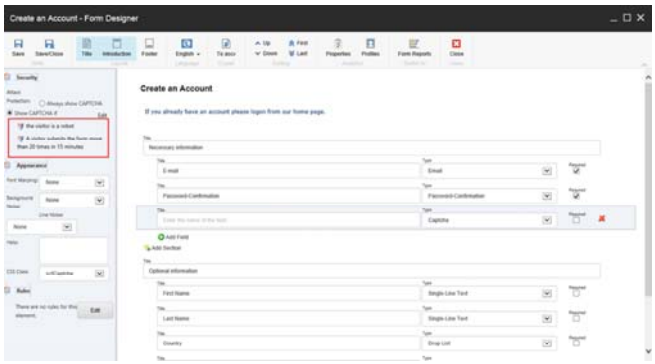
3. In the Robot Attack Protection dialog box, in the Detection Thresholds section, select suspicious form activity detected and enter the maximum number of times a form can be submitted and the time period.



4. To display the form with the CAPTCHA field on a separate page, in the Form Display Page section, select suspicious form activity detected.

5. To display the form with the CAPTCHA field on your custom page:
   - Click Change.
   - Browse to the page you want to display the form with the CAPTCHA field on.
   - Select a placeholder. Contact your Sitecore administrator to find out if this placeholder is enabled using the Select Placeholders wizard.
6. Click OK to close the Robot Attack Protection dialog box.
7. Click Save or Save/Close to save the form changes.

When you have selected suspicious form activity detected, this information is displayed in the Security section along with the time limits that you configured:



Note

When you select suspicious form activity detected, the visitor is a robot condition is automatically enabled.

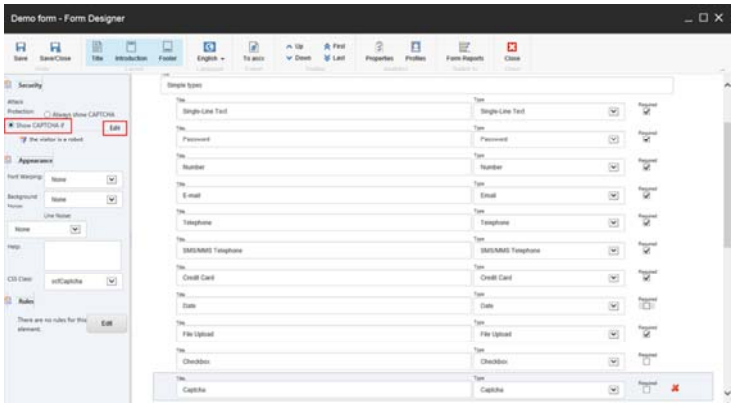
If the form is submitted more than the specified limit of times, the form with the CAPTCHA field is displayed.

Configure a warning email notification

If a robot submitted a form or any of the specified thresholds are exceeded, the Web Forms for Marketers module can send an email notification with a predefined text to specified recipients.

To activate email notifications about robot attacks:

1. In the Form Designer, select the CAPTCHA field.
2. In the Security section, select Show CAPTCHA if and click Edit.



3. In the Robot Attack Protection dialog box, on the Warning Email tab, enter:
   - In the To and CC fields, the recipients' email addresses.
   - In the Subject field, the subject of the email message.
   - In the Message Body field, the text of the email notification.

4. Click OK to close the Robot Attack Protection dialog box.
5. Click Save or Save/Close to save the form changes.

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

Field hierarchy in the Web Forms for Marketers module

If the predefined controls do not meet your specific needs, you can create your own control, for example, to modify the behavior or special UI elements of a form.

There are two types of fields in the Web Forms for Marketers module:

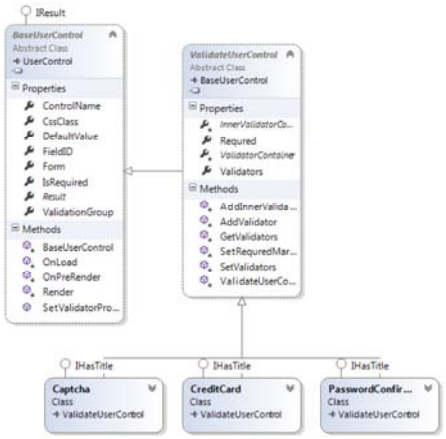
- User control fields – They inherit the `System.Web.UI.UserControl` class. You can use the markup of these fields in a separate `.ascx` file. You can change the appearance of these fields without recompiling assemblies.
- Web control fields – They inherit the `System.Web.UI.WebControls.WebControl` class. Usually, these fields have a simple structure that does not require any modifications after compiling the assemblies. These types of fields are easier to reuse.

User control fields

In the `Sitecore.Form.UI.UserControls` namespace, the following fields inherit the `UserControl` class.

- *Captcha*
- *CreditCard*
- *PasswordConfirmation*

The following diagram shows the inheritance chart for user controls.

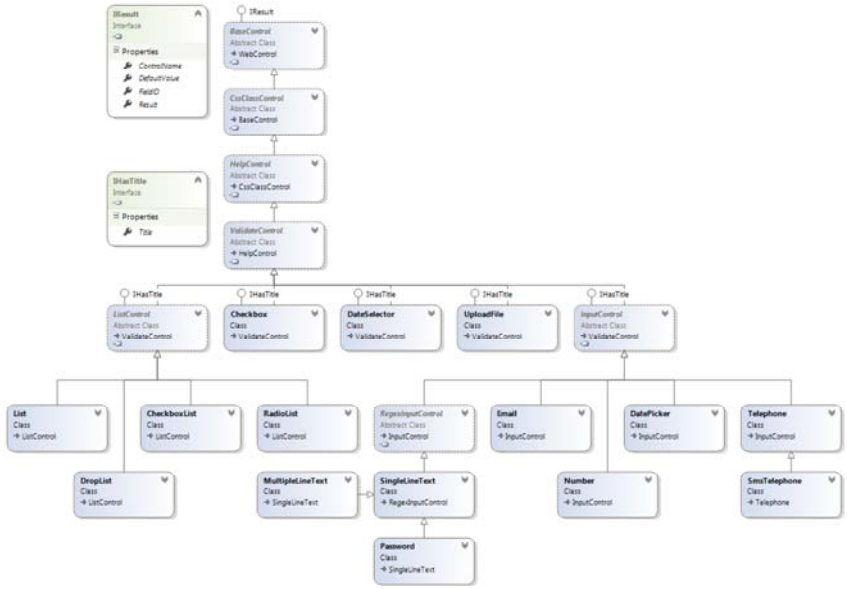


Web control fields

The following list of web control fields are stored in the `Sitecore.Form.Web.UI.Controls` namespace:

- *Checkbox*
- *CheckboxList*
- *DatePicker*
- *DateSelector*
- *DropList*
- *Email*
- *Label*
- *List*
- *MultipleLineText*
- *Number*
- *Password*
- *RadioList*
- *SingleLineText*
- *SmsTelephone*
- *Telephone*
- *UploadFile*

The following class diagram shows the inheritance chart for web controls.



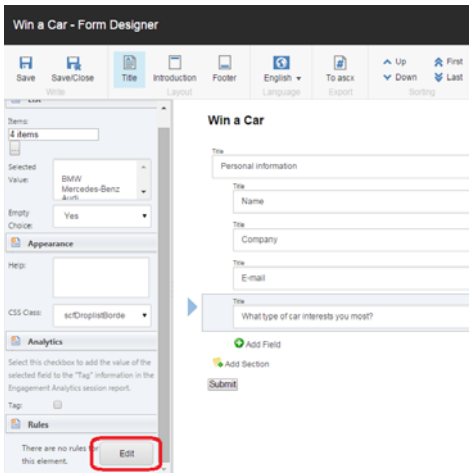
Hide a web form field

In certain situations, you may want to hide a web form field when the necessary conditions are met.

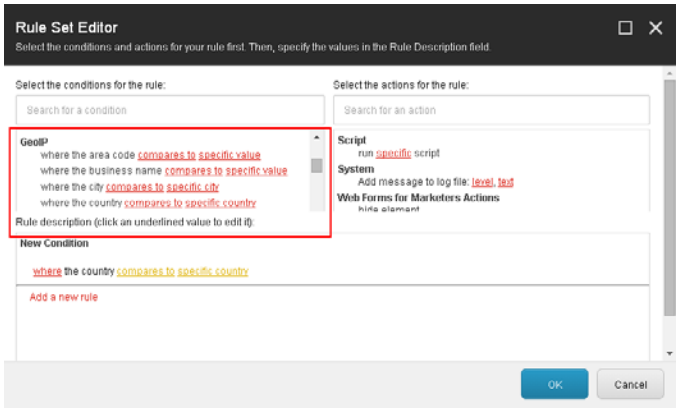
For example, a promotional campaign can differ from country to country and you might want to hide a particular choice in a field on your form. You do this by creating two identical fields with different list items and then hide one of them depending on the geographical location of the site visitor's IP address.

To configure a rule that hides a particular field:

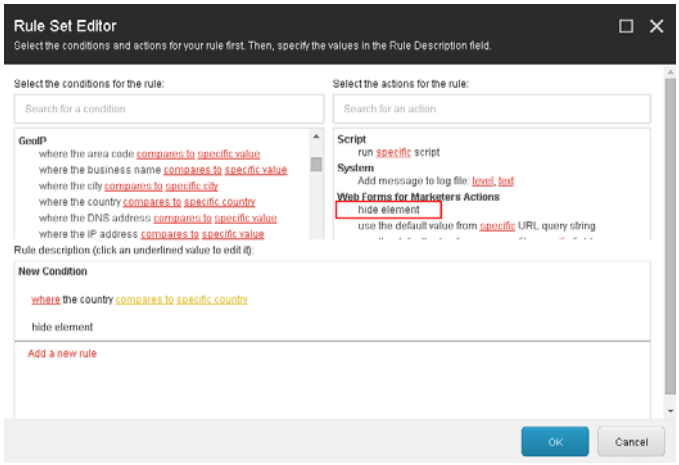
1. In the Form Designer, click the relevant field, and in the left pane, click Edit in the Rules section.



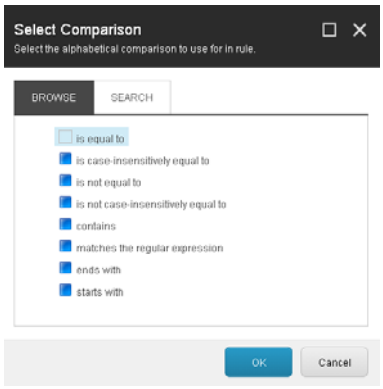
2. In the Rule Set Editor dialog box, in the Select the conditions for the rule field, in the GeoIP section, select the relevant rule, for example, *where the country compares to specific country* condition. This condition appears in the New Condition section.



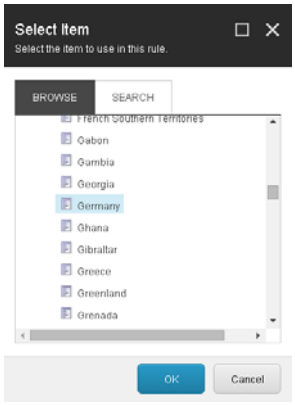
3. In the Select the actions for the rule field, select the relevant action for the rule, for example, *hide element* action. This action appears in the New Condition section.



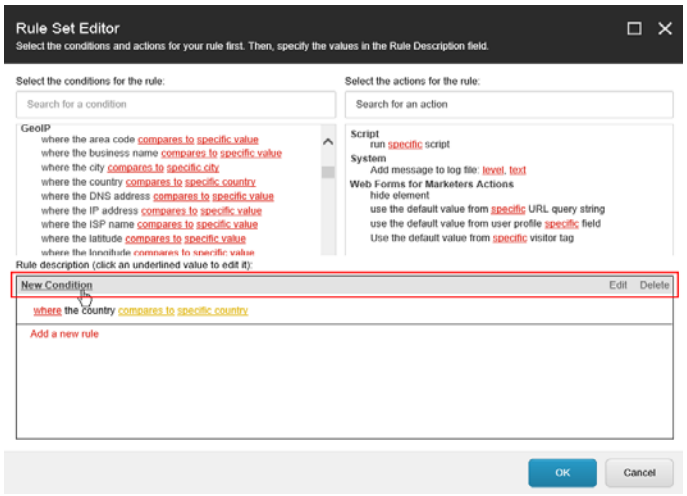
4. The actions for the rule can be configured under the New Condition section. For example, click *compares to* and in the Select Comparison dialog box, change it to *is equal to*. Click OK.



5. In the New Condition section, click the relevant rule, for example, *specific country*, and in the Select Item dialog box, select the relevant item to use, for example, *Germany*, and then click OK.



6. In the Rule Set Editor dialog box, move the mouse to the top of the New Condition field to display the Edit and Delete buttons. Click Edit.



7. In the Message dialog box enter the new name for this condition, for example, *Car type*, and then click OK.
8. In the Rule Set Editor dialog box, click OK and Save or Save/Close the changes in the Form Designer.

Prevent users from uploading large files in web forms

In the Web Forms for Marketers module, you can prevent users from using the File Upload form field to upload files larger than 10 MB.

To prevent users from uploading large files using the File Upload field:

1. In Visual Studio, create a new processor class specifying the file size limit:

```
using Sitecore.Form.Core.Pipelines.FormUploadFile;

  public class UploadingLimitation

  {

    public void Process(FormUploadFileArgs args)

    {
```

```
        int size = 10485760; // == 10 Mb
        if (args.File.Data.Length > size)
        {
          Sitecore.Diagnostics.Log.Info(string.Format("User {0} tried to upload a file larger than 10 Mb. The file name is {1}",
            Sitecore.Context.User.Name,
            args.File.FileName), this);
          args.AbortPipeline();
        }
      }
    }
```

2. Register the new processor in the *Sitecore.Forms.config* file:

```
<formUploadFile>
        <processor type="YourNamespace. UploadingLimitation, YourAssemblyName"/>
        …
</formUploadFile>
```

After you implement the solution, users cannot upload files larger than 10 MB, and the corresponding message is saved to the log files.

Show or hide a form field depending on other field values

For ASP.NET (non-MVC) forms, you can show or hide a form field depending on the values in other fields. For example, when a customer fills out a form, the State field appears in the web form if the Country field value is USA.

To do this, as a prerequisite, you must create a form that contains a Single-line text field and a Checkbox field, and then you must export the form to an .ascx file.

This topic describes how to:

- Create a sublayout item
- Assign a sublayout item to a content item
- Add a script to a form

Create a sublayout item

Sublayouts in Sitecore apply to non-MVC (ASP.NET) web forms and they are composed of a definition item and an `.ascx` control.

After you have exported your web form to an `.ascx` file, you must create a sublayout item in Sitecore and associate the file with the item.

To create a sublayout item:

1. In the Content Editor, in the content tree, click the */sitecore/Layout/Sublayouts* folder, and in the right pane, click Sublayout.

   Note

   It is best practice to create a new folder for your custom sublayouts within the */sitecore/Layout/Sublayouts* folder. However, the location of the */Sublayouts* folder can be different, depending on your Sitecore configuration.

2. In the Name dialog box, enter the name of the `.ascx` file that you created when you exported the web form, and then click Next.
3. In the Location dialog box, click the folder in the Sitecore content tree where you want to store your sublayout item, and then click Next.
4. In the File Location dialog box, click the folder in your computer where you want to store the new sublayout file, and then click Create.
5. Click Close.

Assign a sublayout item to a content item

After you have created a sublayout item, you must assign it to a content item.

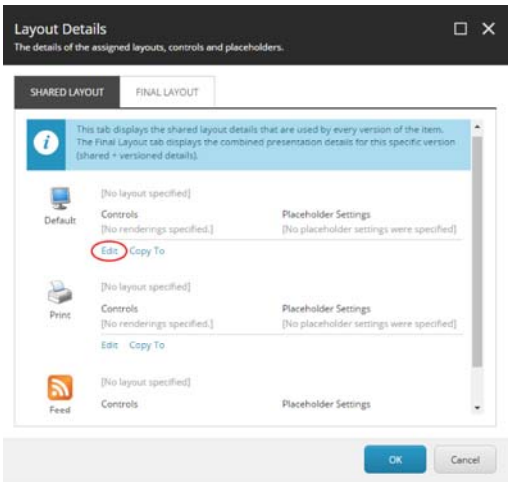
To add a sublayout item to a content item:

1. In the command prompt, copy the `.ascx` file that you exported your form to into the folder where you store your custom sublayout files.

   Note

   When you are asked to *Skip* or *Replace/Overwrite* the file in the destination folder, you must select to *Replace/Overwrite* the file.

2. In the Content Editor, in the content tree, navigate to the location of the relevant sublayout item, and on the ribbon, on the Presentation tab, in the Layout group, click Details.
3. In the Layout Details dialog box, click either the Shared Layout or Final Layout tab.
4. In the Controls section for your relevant device, click Edit.



5. In the Device Editor dialog box, in the left pane, click the Controls tab, and then click Add.
6. In the Select a Rendering dialog box, navigate to the folder in the Sitecore content tree where you stored your sublayout item, and click the rendering that you want to use.
7. In the Add to Placeholder field, enter the name of the placeholder where you want to assign the sublayout, for example, *main*, and click Select.
8. Click OK and in the Layout Details dialog box, click OK again.

Add a script to a form

To change the behavior of UI elements on an exported form, for example, to make their visible state dependent on each other, you must add the appropriate script to the form's `.ascx` file.

To add a script to the relevant form:

1. In a text editor, open the exported `.ascx` file, and copy to your clipboard the value of the ID attributes of the `<input id>` tags of the Single-line text and Checkbox fields. For example:
   - Checkbox field ID:

   ```
   WebUserControl1_field_F2ACADD39B3C46A4A3036C63C0D60C3C
   ```

   - Text field ID:

   ```
   WebUserControl1_field_5339820707D14FAC88D66DCC8F81EB01
   ```

2. Create a client script that shows or hides the text field depending on the selected check box value.
3. Paste the values of the ID attributes of the `<input id>` tags from step 1 in this procedure to the `getElementById` method call.

   For example:

   ```
   <script type="text/javascript">
     var checkbox = document.getElementById("WebUserControl1_field_F2ACADD39B3C46A4A3036C63C0D60C3C");
     var textbox = document.getElementById("WebUserControl1_field_5339820707D14FAC88D66DCC8F81EB01").parentNode.parentNode;
     textbox.style.display = "none";
     checkbox.onclick = function () {
       if (checkbox.checked) {
         textbox.style.display = "block";
       }
       else {
         textbox.style.display = "none";
       }
     }
   </script>
   ```

4. Add the client script to the end of the `.ascx` file and save your changes.

The form you created can now show or hide a form field depending on the value of another field.

The field types in web forms

The Web Forms for Marketers module enables you to create web forms. A form consists of fields, and each field consists of a field name and field type.

In the module, the field type determines how a field is rendered in a form, as well as the number of settings in the Form Designer, client and server validations and how a field value is processed.

Field types are divided into:

Note

Web Forms for Marketers also lets you create your own custom field types.

All the form field types and their settings are stored under the `/sitecore/System/Modules/Web Forms for Marketers/Settings/Field Types` item and are based on the `/sitecore/Templates/Web Forms for Marketers/Field Type` template.

Simple types

Field type	Description
Single-line text	Enter one line of text in this field. The field length is limited to 255 characters by default.
Multi-line text	Enter multiple lines of text in this field. The number of characters is limited to 512 by default.
Password	Enter a password in this text field. All the characters you enter in this field are masked.
Email	Enter email addresses in this field. It checks that it is a valid email address that includes an @ and . as characters, as well as the length of the email server domain.
Telephone	Enter telephone numbers in this number. This field also accepts spaces, and +, -, *(*, and *)* characters.
SMS/MMS telephone	Enter telephone numbers that you can send SMSs and MMSs by typing numerals and the + character in this field. The + character can only be used as the first character in the field. This field is compatible with the *Send SMS* and *Send MMS* save actions.
Number	Enter numerals in this field.
Date	Type dates in this field.
Date picker	Select a date from the calendar in this field.
Checkbox	Display a check box that allows users to enable or disable a condition.
File upload	This field displays the Choose File button that opens up a dialog box where users can select a file to upload to a server.

List types

Field Type	Description
Drop list	Select a single option from a list.
List box	Display a list box that allows you to select one or more items.
Radio list	Display a group of options in a radio list using this field.
Checkbox list	Display a group of check boxes, enabling users to select one or more check boxes from the list.

Complex

Field Type	Description
CAPTCHA	This field prevents robots from registering on websites. This field type is rendered using two fields: an image field and a text confirmation field. The user should enter the characters from the image field in the text field.
Password confirmation	Use this field type to prevent users registering with an incorrect password.

Credit card	This field type contains two fields: a *Password* field and a *Confirm Password* field. These are used to create a password. Every character entered in these fields is masked. Prevent users from entering invalid credit card numbers with this field. This field type contains two fields. One field allows the user to select a credit card type. The other field allows the user to enter a credit card number. Note On MVC forms there is only one field. This field type validates the credit card number with the possible number ranges and combinations allowed by the different types of credit card. You can specify that validation should be based on the Luhn formula or one of the following credit card types: American Express Diners Club Carte Blanche Diners Club International Diners Club US and Canada JCB Maestro MasterCard Solo Switch Visa Visa Electron

Base interface and classes for creating save actions

When you create a custom save action, you must create a class that inherits the `Sitecore.Form.Submit.ISaveAction` interface. This interface contains the *Execute* method that must be implemented in your new class, because it is called by all save actions assigned to a web form. Use this method to implement a save action logic.

The *Execute* method accepts three arguments:

- *ID formid* – the ID of a web form item that the action is assigned to.
- *AdaptedResultList fields* – a list of the *AdaptedResult* classes. Each item in the list provides information about a web form field that is submitted (*Value*, *FieldID*, *FieldName*, *Parameters*).
- *params object[] data* – the first element of the array that contains an analytics session ID.

The Web Forms for Marketers module provides two classes that you can inherit a custom class from:

- Sitecore.Form.Submit.UserBaseAction `class`
- Sitecore.Form.Core.Submit.AuditSaveAction `class`

Configure a default save action

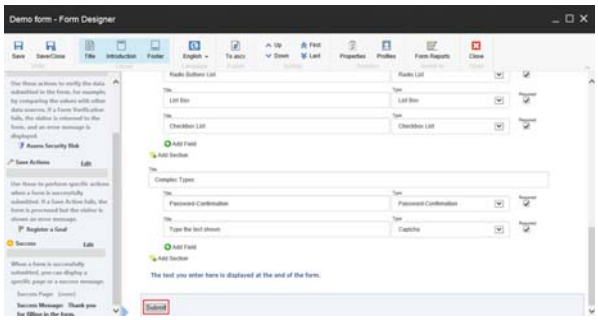
The Web Forms for Marketers module lets you assign actions to a web form that will be performed when all form verifications are successfully completed and the visitor submits the form. These are called save actions and are very similar to form verifications, but will not return the visitor to the form if they fail.

The settings for these save actions are described in the following sections:

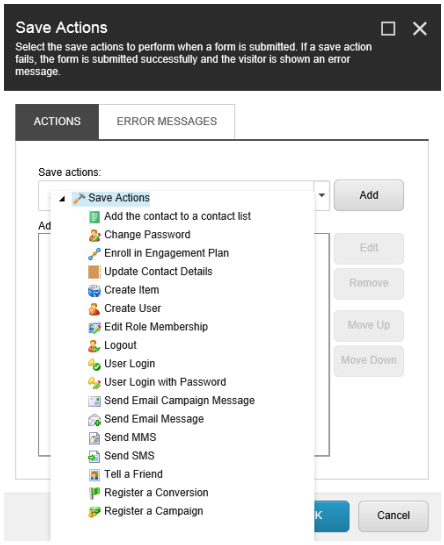
- Create User
- Create Item
- Edit Role Membership

To configure a default save action:

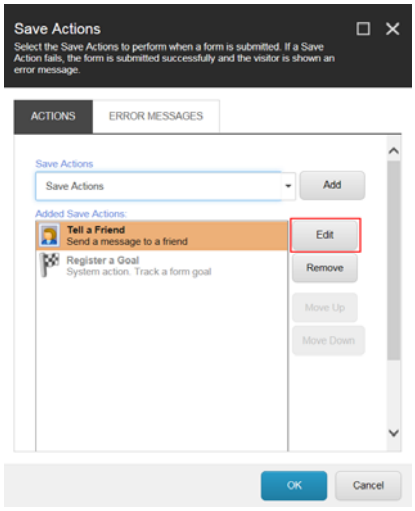
1. Open the relevant form, and in the Form Designer, click the Submit button on the web form.



2. In the left pane, click Save Actions.
3. In the Save Actions dialog, in the Save Actions field, select the relevant save action and click Add.



4. In the Added Save Actions section, select the relevant save action, for example *Tell a Friend* and then click Edit to open the wizard for this save action.



5. In the wizard, fill in the relevant fields.

   Note

   Certain save actions modify user profiles automatically. In the wizard, in the Save Audit Information to field, you can specify where you want to register audit information. It can be useful to register the fields that have been changed, and what the previous values were. Default save actions can register this type of audit information in a selected field in the user profile.

   You can save information to fields of the following types: Rich Text, html, text, Multi-Line Text, Single-Line Text and memo. The *Don't Save* option is selected by default, which means no information about field changes is saved.

Create user

The Create User save action creates a new Sitecore user based on the information that a website visitor provides. This save action also checks whether the user already exists and, if the user does not exist, a new user is created. If the user does exist, the user information is updated.

Basic User Information tab

Field	Description
User name	Select either *Your name* or *Friends e-mail* to specify the user name. The recommended setting is *Friends e-mail.* Note The User Name field is required. When you select *Email*, you make the user name unique. The *localpart_at_domain_dot_topleveldomain* format is used. For example, a user with the *example@domain.com* email address is created with the *example_at_domain_com* user name. The value of this email is also used as the new user's email address (in standard email format).
User Password	Assign a password for the newly created user. By default, *Random Password* is selected. This option randomly generates a password. You can also select the value of a password field in the form, or leave the password blank.
Domain	Specify the domain in which the user is created. You can choose between all the domains available in your Sitecore installation. The *extranet* domain is selected by default.
Associate new user with this visitor	Select to link the visitor's browser, IP, and geographical information with the newly created user. This is useful if the Sitecore security model is used in conjunction with an external system, such as a CRM system.

Additional User Properties tab

Field	Description
Overwrite user field if it already contains a value	Select to make all the selected Form Fields overwrite the corresponding User Profile fields if the user already exists.
Form Field	Specify the field on the form that you want to map to the user profile that you specify in the corresponding User Profile column. This includes fields that may be mapped from other systems using a security provider.
User Profile	Specify the user profile that you want to map the form field to.
Save Audit information to	Specify where you want to register audit information. The *Don't Save* option is selected by default, which means no information about field changes is saved.

Create item

The Create Item save action creates a new item in the content tree every time a visitor clicks Submit on a form.

Select Template dialog

Field	Description
Template	Select the template that you want the items to be based on.
Destination	Select the folder where you want to store the items.

Map the Form Fields dialog

Field	Description
Show Standard Fields	Select to populate the drop-down list with standard fields that you can select.
Mapping fields	Specify which values will be stored in which form fields for all future items that use the template.

Edit Role Membership

The Edit Role Membership save action adds a Sitecore user to a role, or removes the user from a role. If the user does not exist, the user is created as for the Create User save action. The user is created with a randomly generated password.

Edit Role Membership wizard

Field	Description
User Name	Select a relevant value from the drop-down list.
Domain	Select a relevant value from the drop-down list.
Associate existing user with this visitor	Select to link the visitor's browser, IP, and geographical information with the existing user.
Change Role Membership	To edit the role membership only if a specified check box or list value is selected in the form on the web page, select the relevant check box or list field in the drop-down list. The default is *Always*, which means that the save action always edits the role membership of the user.
Add User to Role	To change the roles that the user is a member of click Edit, and in the Edit User Roles dialog box, select the roles that you want to make the user a member of and click Add.
Remove User from Role	Click Edit to open the Edit User Roles dialog box and select the roles that you want to remove the user from, and then click Remove.
Save Audit information to	Specify where you want to register audit information. The *Don't Save* option is selected by default, which means no information about field changes is saved.

Create a new save action

A save action is the second of three submit actions that are performed when a website visitor submits a web form. A save action is performed after all verification actions are successfully completed.

There are several default save actions that are part of the Web Forms for Marketers module. However, you can also create your own custom save actions by using the base interface and classes to create a new class.

To create a custom save action, you must create a class that inherits the `Sitecore.Form.Submit.ISaveAction` interface. This interface contains the `Execute` method that must be implemented in your new class because it is called by all the save actions that are assigned to a web form.

To create a save action:

1. Create a new project in Visual Studio, for example, `Sitecore.Forms.Sample`
2. Add a new reference to the `Sitecore.Forms.Core` assembly.
3. Create a new class that inherits the Sitecore.Form.Submit.ISaveAction interface.

   For example:

```
using Sitecore.Data;
using Sitecore.Diagnostics;
using Sitecore.Form.Core.Client.Data.Submit;
using Sitecore.Form.Core.Controls.Data;
using Sitecore.Form.Submit;
using Sitecore.Security.Authentication;
namespace Sitecore.Forms.Sample
{
   /// <summary>
   /// Login action
   /// </summary>
   public class LoginAction : ISaveAction
   {
      #region Methods
      /// <summary>
      /// Initializes a new instance of the <see cref="LoginAction"/> class.
      /// </summary>
```

```
        public LoginAction()
        {
           this.DefaultDomain = "sitecore";
        }
        /// <summary>
        /// Executes the login action.
        /// </summary>
        /// <param name="formid">The form id.</param>
        /// <param name="fields">The fields of the form.</param>
        /// <param name="data">The custom data.</param>
        public void Execute(ID formid, AdaptedResultList fields, params object[] data)
        {
           AdaptedControlResult login = fields.GetEntry(this.Login, "Login");
           AdaptedControlResult password = fields.GetEntry(this.Password, "Password");
           Assert.ArgumentNotNull(login, "You should fill in the 'Login' field.");
           Assert.ArgumentNotNull(password, "You should fill in the 'Password' field.");
           string userName = login.Value;
           Assert.ArgumentNotNullOrEmpty(userName, "The 'Login' field can't be empty");
           if (!userName.Contains("\\"))
           {
              userName = string.Join("\\", new[] { this.DefaultDomain, userName });
           }
           AuthenticationManager.Login(userName, password.Value, true);
        }
        #endregion
        #region Properties
        /// <summary>
        /// Gets or sets the default domain.
        /// </summary>
        /// <value>The default domain.</value>
        public string DefaultDomain { get; set; }
        /// <summary>
        /// Gets or sets the login field.
        /// </summary>
        public string Login { get; set; }
        /// <summary>
        /// Gets or sets the password field.
        /// </summary>
        public string Password { get; set; }
        #endregion
    }
}
```

4. Compile the class to an assembly and place it in your website's `bin` folder.
5. In the content tree, navigate to the folder *sitecore/System/Modules/Web Forms for Marketers/Settings/Actions/Save Actions* and in the right-hand pane, create a new item by clicking the Save Action button. In the Message dialog, enter a name for the new item, and then click OK.
6. In the Message dialog box, enter a name for the new item and click OK.
7. In the new save action item, fill in the Assembly and Class fields by entering the name of the previously created assembly and class.

   You can further customize the save action by modifying the Parameters and Localized Parameters field.

8. Click Save when you are finished.

Create an Action Editor

An Action Editor is a dialog box that lets you set some of the parameters of a save action. You can create an Action Editor for your custom save action.

Before creating an Action Editor, refer to the Sitecore Developer Network website for information about XML controls and an XML application.

The Action Editor uses the following working logic:

- It reads save action parameters in the `OnLoad` method.
- It shows save action parameters to the user.
- After the user has edited the parameters, it saves them to the Save Actions field of the web form using the `OnOk` method.

Before implementing the Action Editor, you should familiarize yourself with the following code tips:

- To get current save action parameters:

```
string params=HttpContext.Current.Session[Sitecore.Web.WebUtil.GetQueryString("params")] as string;
NameValueCollection nvParams=ParametersUtil.XmlToNameValueCollection(params);
```

- To save parameter values, override the `OnOk` method:

```
protected override void OnOK(object sender, EventArgs args)
    {
      string str3 = ParametersUtil.NameValueCollectionToXml(this.nvParams ?? new NameValueCollection());
      if (str3.Length == 0)
      {
        str3 = "-";
      }
      SheerResponse.SetDialogValue(str3);
      base.OnOK(sender, args);
    }
```

- To get the current web form item ID:

```
Sitecore.Web.WebUtil.GetQueryString("id");
```

- To get the current language:

```
Sitecore.Web.WebUtil.GetQueryString("la", "en");
```

- To get the current Sitecore database:

```
Sitecore.Web.WebUtil.GetQueryString("db");
```

To create an action editor:

1. Create an XML file that contains the dialog layout:

```
<?xml version="1.0" encoding="utf-8" ?>
<control xmlns:def="Definition" xmlns="http://schemas.sitecore.net/Visual-Studio-Intellisense">
   <SimpleEditor>
      <Stylesheet>
        .scfContent {
        padding-top : 15px;
        }

        .scfFieldScope{
        width:100%; margin:7px;
        }

        .scfFieldLabel {
        width:40%;
        }

        .scfFieldSelect {
        width:58%;
        position:absolute;
        right:0;
        margin-right:20px;
        }
      </Stylesheet>

      <FormDialog ID="Dialog" Icon="Software/32x32/step_new.png">
         <CodeBeside Type="Sitecore.Forms.Sample.SimpleEditor,Sitecore.Forms.Sample"/>
        <Border Class="scfContent" Width="100%"  Align="left" Style="overflow:none;">
          <Literal ID="name" Text="Login:" Class="scfFieldLabel"/>
          <Combobox runat="server" ID="login"/>
        </Border>
        <Border Class="scfContent" Width="100%"  Align="left" Style="overflow:none;">
```

```
          <Literal ID="pass" Text="Password:" Class="scfFieldLabel"/>
          <Combobox runat="server" ID="password"/>
        </Border>
         <Border Class="scfContent" Width="100%"  Align="left" Style="overflow:none;">
           <Literal ID="domLiteral" Text="Default domain:" Class="scfFieldLabel"/>
           <Combobox runat="server" ID="domain"/>
         </Border>
      </FormDialog>
   </SimpleEditor>
</control>
```

2. Create an action editor class based on the `Sitecore.Web.UI.Pages.DialogForm`. Specify the created class as `CodeBeside` in the XML file. The following code sample is for the action editor class:

```
public class SimpleEditor : DialogForm
{
    protected Combobox domain;
    protected Combobox login;
    protected Combobox password;
    private NameValueCollection nvParams;
    protected override void OnLoad(EventArgs e)
    {
      base.OnLoad(e);
      if (!Context.ClientPage.IsEvent)
      {
        foreach (Domain d in DomainManager.GetDomains())
        {
          Sitecore.Web.UI.HtmlControls.ListItem item = new Web.UI.HtmlControls.ListItem();
          item.Value = d.Name;
          item.Header = d.Name;
          domain.Controls.Add(item);
        }
        foreach (FieldItem f in this.CurrentForm.Fields)
        {
          Sitecore.Web.UI.HtmlControls.ListItem item = new Web.UI.HtmlControls.ListItem();
          item.Value = f.ID.ToString();
          item.Header = f.DisplayName;
          login.Controls.Add(item);
          item = new Web.UI.HtmlControls.ListItem();
          item.Value = f.ID.ToString();
          item.Header = f.DisplayName;
          password.Controls.Add(item);
        }
        domain.Value = this.GetValueByKey("DefaultDomain") ?? string.Empty;
        login.Value = this.GetValueByKey("Login") ?? string.Empty;
        password.Value=this.GetValueByKey("Password") ??string.Empty;
      }
    }
    protected void SaveValues()
    {
      this.SetValue("DefaultDomain", domain.Value);
      this.SetValue("Login", login.Value);
      this.SetValue("Password", password.Value);
    }
    protected override void OnOK(object sender, EventArgs args)
    {
      this.SaveValues();
      string str3 = ParametersUtil.NameValueCollectionToXml(this.nvParams ?? new NameValueCollection());
      if (str3.Length == 0)
      {
```

```
    str3 = "-";
  }
  SheerResponse.SetDialogValue(str3);
  base.OnOK(sender, args);
}
public void SetValue(string key, string value)
{
  if (this.nvParams == null)
  {
    this.nvParams = ParametersUtil.XmlToNameValueCollection(this.Params);
  }
  this.nvParams[key] = value;
}
public string GetValueByKey(string key)
{
  if (this.nvParams == null)
  {
    this.nvParams = ParametersUtil.XmlToNameValueCollection(this.Params);
  }
  return (this.nvParams[key] ?? string.Empty);
}
public string Params
{
  get
  {
    return (HttpContext.Current.Session[Sitecore.Web.WebUtil.GetQueryString("params")] as string);
  }
}

public string CurrentID
{
  get
  {
    return Sitecore.Web.WebUtil.GetQueryString("id");
  }
}
public FormItem CurrentForm
{
  get
  {
    if (!string.IsNullOrEmpty(this.CurrentID))
    {
      Item innerItem = this.CurrentDatabase.GetItem(this.CurrentID, this.CurrentLanguage);
      if (innerItem != null)
      {
        return new FormItem(innerItem);
      }
    }
    return null;
  }
}
public virtual Database CurrentDatabase
{
  get
  {
    return Factory.GetDatabase(Sitecore.Web.WebUtil.GetQueryString("db"));
  }
```

```
    }
    public virtual Language CurrentLanguage
    {
      get
      {
        return Language.Parse(Sitecore.Web.WebUtil.GetQueryString("la", "en"));
      }
    }
  }
```

3. In the appropriate Save Action item, in the Editor field, specify the editor that you have created.

Customize the Send Email Message save action

The Send Email Message save action sends an e-mail message every time a visitor clicks the Submit button on a web form. You can configure an email message in this save action using the `ProcessMessage` pipeline.

To configure an email message:

1. Create a processor class using the following sample code:

```
using Sitecore.Form.Core.Pipelines.ProcessMessage;
// This processor adds a note to the end of the email body
  public class AddTextToBody
  {
    public void Process(ProcessMessageArgs args)
    {
      string additionalText = "<p>This message was sent using the Sitecore Web Forms for Marketers module.</p>";
      args.Mail.Append(additionalText);
      /*
       * it's also possible to modify SUBJECT, TO, CC and BCC message fields
       * args.Subject.Append(" subject text");
       * args.To.Append("; secondrecipient@mail.net");
       * args.CC.Append("; secondrecipient@mail.net");
       * args.BCC.Append("; secondrecipient@mail.net");
       * args.From = "sender@mail.net";
       */
    }
  }
```

2. Register the new processor in the `Sitecore.Forms.config` file, before the *Sitecore.Form.Core.Pipelines.ProcessMessage.ProcessMessage* processor:

```
      <processMessage>
        …
        <processor type="YourNamespace.AddTextToBody,YourAssemblyName" />
        <processor type="Sitecore.Form.Core.Pipelines.ProcessMessage.ProcessMessage, Sitecore.Forms.Core" method="SendEmail"/>
      </processMessage>
```


Modify a save action using a field condition

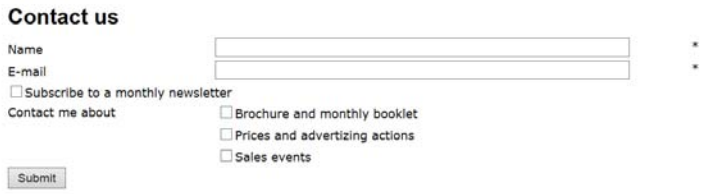
When a user successfully submits a web form, you can specify actions that should occur. These actions are called save actions. You can modify a save action so that it is only executed if the visitor selects certain values in the form. You use field conditions to do this. The following save actions contain field conditions:

- Edit Role Membership (Change Role Membership field)
- Register a Conversion (Register Conversion field)
- Send SMS (Send Message field)
- Send MMS (Send Message field)
- Enroll in Engagement Plan
- Add Contact to Contact List

Field conditions contain all form list fields and check boxes. You select the condition that must be fulfilled before a save action is performed.

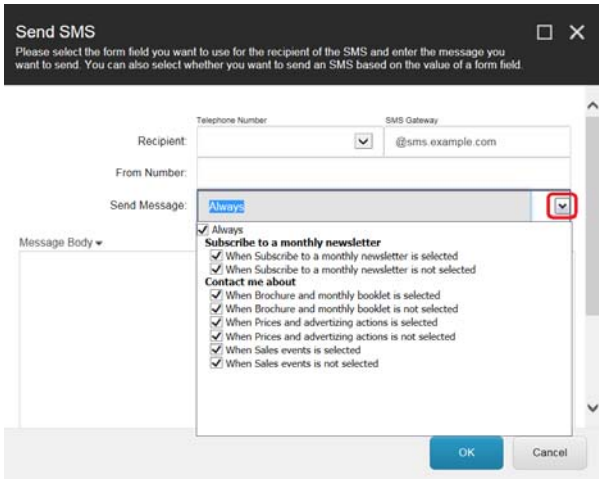
To modify a save action when certain conditions are fulfilled:

1. In the Form Designer, open the relevant form and add the relevant save action. For example, if you want an SMS to be sent to a visitor when they select the Subscribe to a monthly newsletter, and the Sales events check boxes on your Contact Us form, add the *Send SMS* save action.



2. In the dialog that appears, select the condition that must be fulfilled before the save action is performed.

   For example, in the Send SMS dialog, in the Send Message field, click the arrow to open the list of form list items and check boxes:



   Clear the Always check box, and then select the When Subscribe to a monthly newsletter is selected and the When Sales events is selected check boxes.

3. When you have finished filling in the fields in the dialog, save your changes.

   In this example, when a visitor selects the Subscribe to a monthly newsletter and the Sales events check boxes, the Web Forms for Marketers module automatically sends an SMS message to the visitor's mobile phone.

Security actions

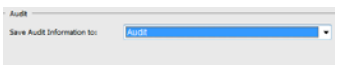
Security actions are selected default save actions that create or edit users or roles in Sitecore's security model.

Security actions include the following default save actions:

- Create User
- Edit Role Membership
- User Login
- User Login with Password
- User Logout
- Change password

Because security actions can affect user information, it is useful to have the ability to register audit information in the user profile to record what actions have been performed.

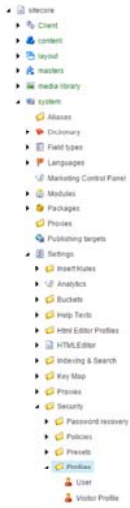
All of the security actions contain the Save Audit Information to: drop-down list, which lists the field in a user profile to which audit information can be written.



By default, all rich text, html, text, memo, multi-line text, and single-line text fields can be used to register audit information. The field types in which audit information can be registered, can be configured using the *WFM.AuditAllowedTypes* setting in the `Sitecore.Forms.config` file:

```
<setting name="WFM.AuditAllowedTypes" value="|Rich Text|html|text|Multi-Line Text|Single-Line Text|memo|" />
```

All user profiles are items in the Core database in the */sitecore/System/Settings/Security/Profiles* folder.



The *Visitor* profile is used by default. Each form item has a reference to this user profile in the User Profile field.

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

Send SMS/MMS in web forms using a custom processor

In the Web Forms for Marketers module, you can send SMS and MMS messages using the *Send SMS* and *Send MMS* save actions respectively. The *Send MMS* and the *Send SMS* save actions delegate sending messages to an MMS/SMS gateway through an SMTP server.

You can use the `processMessage` pipeline to change this behavior and send messages, for example, through a third-party paid web service.

To send messages through a third party web service using the `processMessage` pipeline:

1. In Visual Studio, in the `Sitecore.Form.Core.Pipelines.ProcessMessage` namespace, create a new processor using the following sample code:

```
namespace Sitecore.Form.Core.Pipelines.ProcessMessage
{
  using System.IO;
  using System.Net;
  public class SendSMSorMMS
  {
    public void Process(ProcessMessageArgs args)
    {
      if (args.MessageType == MessageType.MMS || args.MessageType == MessageType.SMS)
      {
        WebClient wc = new WebClient();
        wc.Credentials = (NetworkCredential)args.Credentials;
        wc.QueryString.Add("sendto", args.Recipient);
        wc.QueryString.Add("message", args.Mail.ToString());
        if (!string.IsNullOrEmpty(args.From))
        {
          wc.QueryString.Add("from", args.From);
        }
        using (Stream responseStream = wc.OpenRead("https://3rdparty.smsormms.com/"))
        {
          using (StreamReader responseReader = new StreamReader(responseStream))
          {
            responseReader.ReadToEnd();
            responseReader.Close();
            responseStream.Close();
          }
        }
      }
    }
  }
```

```
}
```

2. In the *Sitecore.Forms.config* file, register the new processor:

```
<processMessage>
  <processor type="Sitecore.Form.Core.Pipelines.ProcessMessage.ProcessMessage, Sitecore.Forms.Core" method="ExpandLinks"/>
  <processor type="Sitecore.Form.Core.Pipelines.ProcessMessage.ProcessMessage, Sitecore.Forms.Core" method="ExpandTokens"/>
  <processor type="Sitecore.Form.Core.Pipelines.ProcessMessage.ProcessMessage, Sitecore.Forms.Core" method="AddHostToItemLi
  <processor type="Sitecore.Form.Core.Pipelines.ProcessMessage.ProcessMessage, Sitecore.Forms.Core" method="AddHostToMediaI
  <processor type="Sitecore.Form.Core.Pipelines.ProcessMessage.ProcessMessage, Sitecore.Forms.Core" method="AddAttachments".
  <processor type="Sitecore.Form.Core.Pipelines.ProcessMessage.ProcessMessage, Sitecore.Forms.Core" method="BuildToFromRecip
  <processor type="Sitecore.Form.Core.Pipelines.ProcessMessage.SendSMSorMMS, MyAssembly"/>
  <processor type="Sitecore.Form.Core.Pipelines.ProcessMessage.ProcessMessage, Sitecore.Forms.Core" method="SendEmail"/>
</processMessage>
```

3. From the Sitecore Desktop, open the Form Designer and in the relevant form, add the *Send SMS* or *Send MMS* save action. As an optional step, submit the form to ensure that the custom processor works correctly.

Sitecore.Form.Core.Submit.AuditSaveAction class

The `Sitecore.Form.Core.Submit.AuditSaveAction` class lets you collect audit messages when a save action is executed. It is the base class for the `UserBaseAction` class.

The `Sitecore.Form.Core.Submit.AuditSaveAction` class contains the following methods:

Method	Description
`public void AuditMessage(string message)`	Adds a message to the messages collection (class member).
`public void AuditSkippedField(string takenFrom, string insertTo, string value)`	Adds an entry to the skipped collection (class member). This method can be used to track profile fields that were not overwritten.

Sitecore.Form.Submit.UserBaseAction class

The `Sitecore.Form.Submit.UserBaseAction` class lets you manage users' accounts. The `Sitecore.Form.Submit.UserBaseAction` class contains the following methods:

Method	Description
`public static string Escape(string userName)`	Formats a username (replaces "." with "_dot_" and "@" with "_at_").
`public static string GetFullUserName(string domainName, string userName)`	Ensures that a username contains a domain name.
`protected string GetProfileProperty(UserProfile profile, string profileproperty)`	Gets a profile property value by its name.
`protected string GetUserNameIfExist(string preUserName)`	Returns a username if the user with the specified name exists, otherwise returns null.
`public static string GetValidUserName(string domainName, string userName)`	Returns a valid username based on the specified one.
`protected virtual string ProccessBaseOperations(ID formId, AdaptedResultList fields, bool createIfNotExist)`	Returns a username if the user exists or was created (depending on the `bool createIfNotExist` argument value) by the method.
`protected virtual void UpdateEmail(string userName, string mail)`	Sets the user mail if the specified email is valid.
`protected virtual void UpdateGlobalSession(string userName)`	Updates the analytics global session with the specified user.

`protected virtual void UpdatePassword(ID formID, string userName, string password)`	Creates a user with the specified name and password if none exists. If the Password property value is *blankPassword*, the user is created with a blank password, if the value is *randomPassword*, a random password is generated. This method also sets the profile item ID for the created user.
`protected void UpdateProfileProperty(UserProfile profile, string profileproperty, string propertyValue)`	Sets the profile property value.

The `Sitecore.Form.Submit.UserBaseAction` class contains the following properties:

Property	Description
`public bool AssociateUserWithVisitor { get; set; }`	Indicates whether the analytics global session must be updated with the user information. This property is checked in the *UpdateGlobalSession* method.
`public string AuditField { get; set; }`	It is only used in derived types.
`public string DomainField { get; set; }`	Specifies the domain name.
`public string PasswordField { get; set; }`	Specifies if the password field name can contain special values.
`public string ProfileItemId { get; set; }`	Sets or gets the ID of the user profile item.
`public string UserNameField { get; set; }`	Sets or gets the name of the web form field that contains a username.
`public string UserNameIsEmpty { get; set; }`	Sets or gets the exception message that is thrown if a username is empty.

Specify different SMTP settings for the Send Email Message save action

The Send Email Message save action sends an email every time a visitor clicks the Submit button on a form. The email is sent using the SMTP protocol, and you can specify different SMTP settings for the save action than those that are set by default.

If your Sitecore solution works in a multiserver environment, you must follow the multiserver environment section.

This topic describes how to:

Specify different SMTP settings for the Send Email Message save action

To specify different SMTP settings for the Send Email Message save action:

1. In the content tree, navigate to the *Save Actions* folder *(sitecore/System/Modules/Web Forms for Marketers/Settings/Actions/Save Actions)*, and click the *Send Email Message* item.
2. In the right pane, in the Submit tab, in the Parameters field, enter the relevant code to specify different values from those defined in the `Sitecore.config` file. For example:

   ```
   <Host>smtp-mail.outlook.com</Host><Port>587</Port><EnableSsl>true</EnableSsl><Login>sitecore@outlook.com</Login><Password>xyz1234
   ```

   The following list shows the available parameters that you can use with this save action, and their description:

Parameter	Description
<Host></Host>	Mail server host name
<Port></Port>	Mail server port number
<EnableSsl></EnableSsl>	Use SSL
<Login></Login>	Login to access the mail server
<Password></Password>	Password to access the mail server
<IsBodyHtml></IsBodyHtml>	Email body is in HTML format

<IsIncludeAttachments></IsIncludeAttachments> Email contains attachments

3. If you want to specify credentials for the SMTP server, in the right pane, in the Parameters field, enter the following text and ensure you specify the relevant login and password credentials. For example:

   ```
   <Login>mylogin123</Login><Password>mypassword123</Password>
   ```

4. Click Save to save your changes.

Note

To configure the SMTP settings on an item, in the IIS Manager, ensure that you clear the SMTP email settings.

Specify different SMTP settings for the Send Email Message save action for a multiserver environment

If your Sitecore solution works in a multiserver environment with several Content Delivery (CD) instances, and the Send Email Message save action is configured to run on a CD instance, and each of the CD instances should use its own SMTP settings, you must perform the following steps:

On a Content Delivery (CD) instance

1. On the Content Delivery (CD) instance, in a text editor, open the `Sitecore.config` file.
2. Configure the following SMTP settings: *MailServer*, *MailServerUserName*, *MailServerPassword*.

On a Content Management (CM) instance

1. In the content tree, navigate to the *Save Actions* folder *(sitecore/System/Modules/Web Forms for Marketers/Settings/Actions/Save Actions)* and click the *Send Email Message* item.
2. In the right pane, in the Submit tab, in the Parameters field, delete the `<Host></Host>`, `<Login></Login>` and `<Password><Password>` parameters, if you have any.
3. Click Save, and republish your website.

Specify different SMTP settings for the Send MMS save action

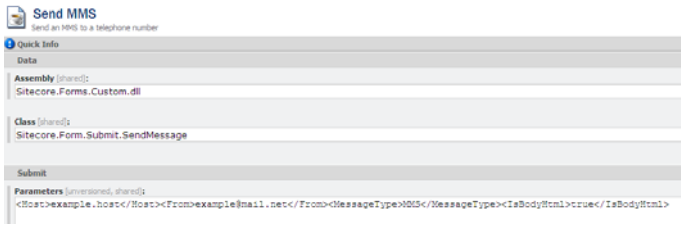
The Send MMS save action sends an MMS message every time a visitor clicks the Submit button of a web form. You can specify SMTP settings for this save action to override the default ones.

To specify different SMTP settings for this action:

1. In the Content Editor, navigate to the folder *sitecore/System/Modules/Web Forms for Marketers/Settings/Actions/Save Actions* and select the *Send MMS* item.
2. In the right pane, in the Submit section, in the Parameters field, set the relevant values.

   For example, to use *Host* and *From* parameters that are different from the ones defined in the `web.config` file, enter the following code into the Parameters field replacing "example" with the relevant host and email address:

   ```
   <Host>example.host</Host><From>example@mail.net</From><MessageType>MMS</MessageType><IsBodyHtml>true</IsBodyHtml>
   ```



3. Click Save.

The default save actions

You can assign actions to web forms after form verifications have successfully taken place and the visitor submits the form. These are called save actions and are similar to form verifications but will not return the visitor to the form if they fail. Some of these default save actions are used to create or edit users or roles in Sitecore's security model. These are referred to as security actions.

The Web Forms for Marketers module contains the following default save actions:

Default Save Action	Description
Add Contact to a Contact list	Adds a contact to a list. A contact is defined based on the context, the lists are defined in the Contact List field. This save action verifies that the context user is logged in. If the user is not logged in, no action is taken.
Change Password	Changes a Sitecore user's password.

	A user is identified based on the values selected in the User Name and Domain fields. This save action first verifies that the user exists. If the user does not exist, no action is taken.
Create User (Security action)	Creates a new Sitecore user based on the information supplied by the visitor in specified form fields. This save action checks whether the user already exists, and if the user does not exist, a new user is created. If the user does exist, the user's information is updated.
Create Item	Creates a new item in the content tree every time a visitor clicks the Submit button on a form.
Edit Role Membership (Security action)	Adds a Sitecore user to a role, or removes the user from a role. If the user does not exist, the user is created with a randomly generated password.
Enroll in Engagement Plan	Places a visitor in a specific state in an engagement plan, after they successfully submit a form. The Enroll the visitor drop-down list contains all the relevant check boxes from the web form that define the condition. You select the conditions that must be met before the visitor is placed in an engagement plan and you select a state in the engagement plan.
Register a Campaign	Registers a visitor as a member of a campaign, after they successfully submit a form. In the Select Campaign dialog box, you select a campaign that you want to associate with the form. Note You cannot associate the web form with an umbrella campaign group.
Register a Conversion	Registers a conversion for the successful completion of an existing goal that you select. The Register Conversion drop-down list contains all the check box and list fields in the web form. By default, the save action always registers a conversion for the selected goal. You can also choose to only register a conversion when a visitor selects a specified check box in the form.
Send Email Campaign Message	Enables you to send email messages using the Email Experience Manager module. This means that you can use the functionality of the Email Experience Manager module and send already created email messages.
Send Email Message	Sends an email message every time a visitor clicks the Submit button on a form. The recipient and body of the email can differ for each form. The email is sent using the SMTP protocol. The item for this action contains the following fields: • To – the addresses of the direct recipients of this email message separated by a semi-colon (;) • CC – the copied (CC) recipients of this email message separated by a semi-colon (;) • Subject – the subject line of this email message • Body – the text of the email message In the Insert Field, you can select from a drop-down list the form field value that you want to insert into the body text. The settings that are defined using this editor are stored in the form item. To add the value from a form field to the To/CC/Subject fields, click the appropriate field name. You can also specify different SMTP settings for the Send Email Message save action.
Send MMS	Sends an MMS message every time a visitor clicks the Submit button. The recipient and body of the MMS message can differ for each form. The item for this action contains the following fields: • Recipient – the phone number and MMS gateway of the recipient of this MMS message. Use the drop-down list to select the form field of the SMS/MMS Telephone type that you want to define the recipient's phone number. • From Number – the phone number that sends the MMS message • Send Message – the condition that should be met to send the MMS message. By default, the save action always sends the MMS message to the selected recipient. You can also choose to send the MMS message only when a specified check box or list value in the form is selected by the visitor. Select the check box or list field in the form from the drop-down list. • Message Body – the text of the MMS message. You can use the Rich Text Editor tools while composing a message. Click Insert Field to add the value from a form field to the MMS message. This exchanges the chosen field with the chosen value when the action is executed. You can also specify different SMTP settings for the Send MMS save action.
Send SMS	Sends an SMS message every time a visitor clicks the Submit button. The recipient and body of the SMS message can differ for each form. The SMS message is sent as plain text. The item for this action contains the following fields: • Recipient – the phone number and SMS gateway of the recipient of the SMS message. Use the drop-down list to select which form field to use for the recipient's phone number. Only SMS/MMS Telephone fields can be used with this action. • From Number – filled in automatically depending on the method used to send the SMS. The default method is not set.

	• Send Message – the condition that should be met before the SMS message is sent. By default the save action always sends the SMS message to the selected recipient. You can also choose to send an SMS message only when a specified check box or list value in the form is selected by the visitor. Select the check box or list field in the form from the drop-down list. • Message Body – the text of the SMS message. Click the arrow next to this field to add the value from a form field to the SMS message.
Tell a Friend	Used by a website visitor to send a predefined email to another person. This web form save action uses the Send Email Message save action. The action uses the values entered in specific form fields to compose the email message.
Update Contact Details	Updates the current contact details, based on the information supplied by the visitor in the fields that are specified in the web form. In the Update Contact Details wizard, the value entered in the field selected in the Form Field column is used as the value in the field selected in the Contact Details column.
User Login (Security action)	Logs in a Sitecore user based on the values entered into the selected form fields, when no password is provided. The save action first verifies that the user exists. If the user does not exist, no action is taken. Select the Associate existing user with this visitor check box to link the visitor's browser, IP, and geographical information with the existing user information.
User Login with Password (Security action)	Logs in a Sitecore user based on the values entered into the selected form fields and checks whether the password is valid. The save action first verifies that the user exists. If the user does not exist, no action is taken. Select the Associate existing user with this visitor check box to link the visitor's browser, IP, and geographical information with the existing user information.
Logout	Logs a Sitecore user out if the form is submitted successfully. This action requires that the Sitecore user is logged in.

The save action item fields

A save action item contains the following configuration fields:

Field	Description	Sample Field Value
Assembly	An assembly name that contains the associated class.	`Sitecore.Forms.Custom.dll`
Class	An associated class name including namespace.	`Sitecore.Form.Submit.SendMessage`

Parameters	A save action can have any number of parameters. In the save action class, parameters are represented as class properties of the string type:
	`public string MyProperty{ get;set;}`
	
	You can specify parameter values in the following format:
	`<parameter name>value</parameter name>`
	
	Specific web form parameters are stored in the web form item and you can edit them in the Form Designer, in the Action Editor.
	
	The parameters in the save action item override the parameters specified in the web form item.
	
	To use a parameter in the action, add a property to a custom action class, and name this property the same as the parameter.
		`<DefaultDomain>extranet</DefaultDomain>`
		`If the action class contains the DefaultDomain property, this property is initialized`
		`/// Gets or sets the default domain./// </summary> /// <value>The default domain.</va`

Localized Parameters	Similar to Parameters. The only difference is that this field is not shared so a parameter can be localized.	`<DefaultDomain>extranet</DefaultDomain>`
		`If the action class contains the DefaultDomain property, this property is initialized`
		`/// Gets or sets the default domain./// </summary> /// <value>The default domain.</va`

Client Action	This field is only used in the staging environment. If this check box is cleared, this	Selected

	save action is transferred from the Slave to the Master server and is performed there. If this check box is selected, a save action is performed on the Slave server.	
Editor	A control that a Sitecore user uses when they edit the parameters for the save action in the Form Designer.	`control:Forms.MappingFields`
QueryString	Additional settings for the editor.	`Fields=Login|Login,Password|Password`

Use save actions to add contacts to a contact list

In the Web Forms for Marketers module, to add a new contact to a contact list, you must first create an empty contact list in the List Manager. You must also create a web form that you want your contacts to submit, which must include the following field types:

- Single-line text field (Name)
- Email field (Email)
- Password field (Password)
- CAPTCHA field (Type the text shown)
- Checkbox field (Subscribe to email)

To verify whether a contact is logged in and to add the contact to a contact list when they submit the form, you must add and configure four save actions in a particular sequence.

This topic describes how to:

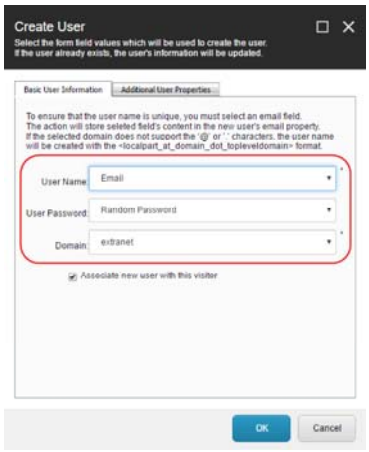
- Add and configure the Create User save action
- Add and configure the User Login save action
- Add and configure the Update Contact Details save action
- Add and configure the Add Contact to Contact List save action

Add and configure the Create User save action

The Create User save action creates a new Sitecore user based on the information supplied by an unidentified contact in specified form fields. However, if the user already exists, this save action updates the user's information.

To add and configure the Create User save action:

1. In the Form Designer, open the form you want your contacts to submit, and on the form, click the Submit button.
2. In the left pane, click Save Actions and in the Save Actions dialog box, in the Save actions drop-down list, click Create User, and then click Add.
3. Click Edit.
4. To map the User Name field to one of your web form fields, in the Create User dialog box, on the Basic User Information tab, click the User Name drop-down list, and click the form field that you want to map the User Name field to.

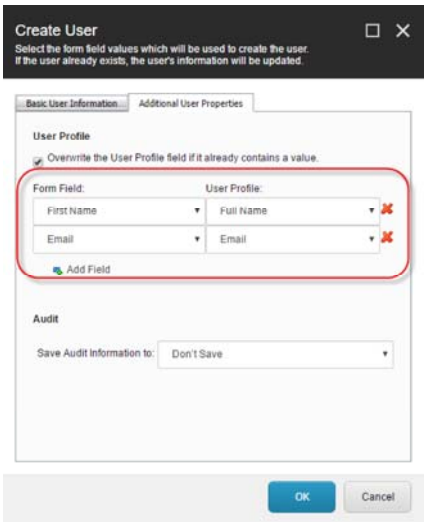


5. Click the Domain drop-down list, and click the relevant domain where you want the user to be created. You can select any of the domains available in your Sitecore installation. The extranet domain is selected by default.
6. If you want to map additional fields on the form to the Sitecore user profile, on the Additional User Properties tab, click Add Field.

   Note

   If you select the Overwrite the User Profile field if it already contains a value check box, and the user already exists, all the form fields selected on the Additional User Properties tab overwrite the corresponding User Profile fields.

7. In the Form Field column, click the drop-down list and click the form field that you want to map, and in the User Profile column, click the drop-down list and click the corresponding Sitecore user profile field. For example:



Note

To specify where you want to register audit information, click the Save Audit Information to drop-down arrow, and select where you want to save it.

By default, Don't save is selected, which means that no information about field changes is saved.

8. Click OK.

   The data entered in the relevant fields is reflected in the User Manager for a newly created user.

Add and configure the User Login save action

After the Create User save action checks whether the user already exists, and either creates a new user or updates the user's information, the User Login save action authenticates the Sitecore user based on the values entered in specific form fields. This save action only works with existing users.

To add and configure the User Login save action:

1. In the Save Actions dialog box, in the Save actions drop-down list, click User Login ,and then click Add.
2. Click Edit.
3. To map the user name to one of your form fields, in the User Login dialog box, in the User Name drop-down list, click the form field that you want to map the user name to. For example: Email.
4. Click the Domain drop-down list, and click the relevant domain. Click OK.

Add and configure the Update Contact Details save action

The Update Contact Details save action updates the current contact details of identified, logged-in contacts based on the information entered in the fields specified on the web form.

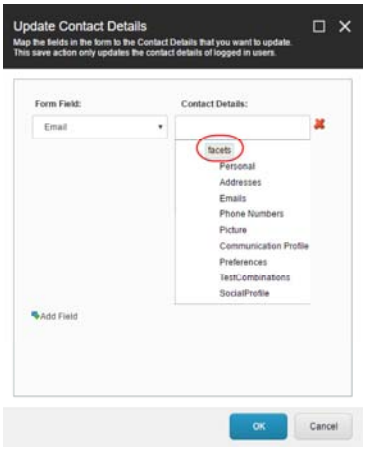
To add and configure the Update Contact Details save action:

1. In the Save Actions dialog box, in the Save actions drop-down list, click Update Contact Details, and then click Add.
2. Click Edit.
3. To update the contact details of an identified, logged-in contact when a form is submitted, in the Update Contact Details dialog box, click the Form Field drop-down list, and click the form field that you want the module to update the details of. For example: Email.

   Note

   If you want to add and map an additional field, click Add Field .

4. To map the selected form field(s) to the corresponding Contact Details field and facet that you want to update, click the Contact Details field, and in the drop-down list that appears, click facets.



5. Click the facet you want to assign to the form field. For example: SmtpAddress.

6. Click OK.

The next time an identified or logged-in visitor submits the form, his or her contact details are updated based on the information supplied in the fields specified on the web form.

Add and configure the Add Contact to Contact List save action

The Add Contact to Contact List save action is the last of the save actions that you must add to the web form so that when an unidentified contact submits the form, the contact is added to the contact list.

To add and configure the Add Contact to Contact List save action:

1. In the Save Actions dialog box, in the Save actions drop-down list, click Add Contact to Contact List, and then click Add.
2. Click Edit.
3. To add the contact to a list when the contact submits the form, in the Contact Lists section, select the relevant contact list check box.
4. To select the conditions that describe when the contact should be added to a contact list, in the Condition field, click the drop-down list and select the relevant condition check boxes.

5. Click OK.
6. Click Save or Save/Close to save your changes.
7. To implement the changes, in the Content Editor, on the ribbon, on the Publish tab, in the Publish group, click Publish.

The next time an unidentified contact fills in and submits the form, the contact is added to the contact list and you can see the new details of the contact in the List Manager.

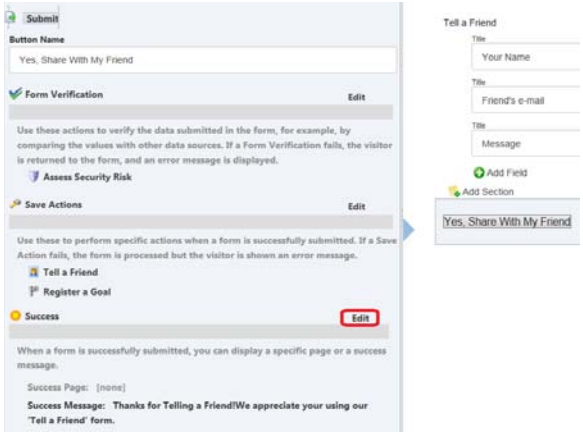
Send feedback about the documentation to docsite@sitecore.net.

Add a success submit action to a form

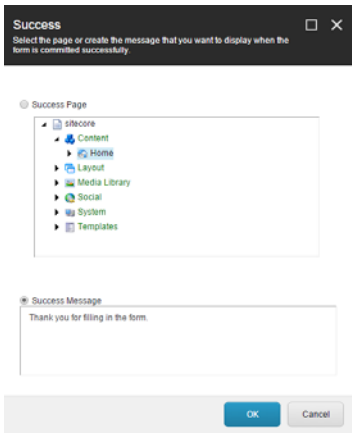
The success submit action is the third and final submit action that you can create for a web form. After a website visitor successfully submits a form, you can add a success submit action that either directs the user to a specific page or displays a success message on the form.

To add a success submit action to a web form:

1. In the Form Designer, on the relevant web form, click the Submit button.
2. On the left pane, in the Success section, click Edit.



3. In the Success dialog box:
   - To redirect the visitor to a page after they have successfully submitted the form, select the Success Page check box and then browse to the relevant webpage.



- Alternatively, to display a message in place of the form, after the visitor successfully submits the form, select the Success Message check box and enter a message in the field.

4. Click OK to save the changes.

Configure a form verification

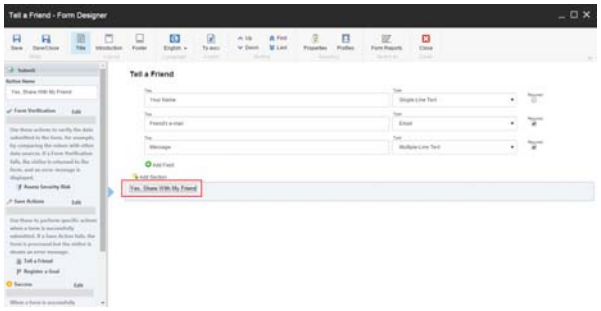
You can use a form verification to check the values that have been entered in one or more fields in a web form. When a user submits a web form, you can set up actions to verify the data submitted in the form.

In the Web Forms for Marketers module, you can configure the following default web form verifications:

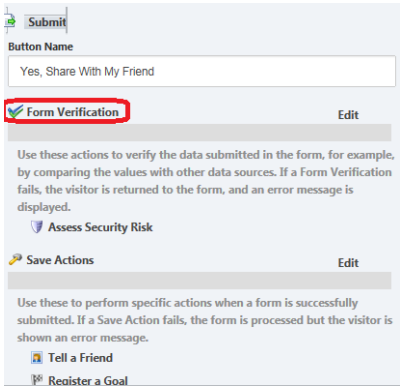
- Assess Security Risk
- Check User and Password
- Is User in Role
- User Exists

To configure a default web form verification:

1. In the relevant web form, open the Form Designer and click the Submit button.



2. In the left pane, click Form Verification.



3. In the Form Verification dialog, in the Form Verification field, click the drop-down arrow and select the form verification that you want to add to the form, and then click Add. The default form verifications are:
   - Assess Security Risk – verifies the information entered in form fields for content that may be malicious. The content could be, for example, executable scripts or similar data. If verification fails, the message “The fields contain content that may present a security risk. Please enter appropriate information” is displayed.

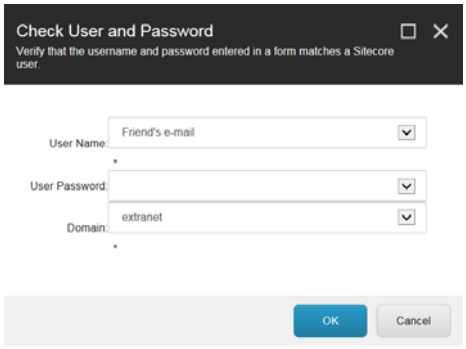
     This action is default for all web forms, and you cannot remove it.

Note

Contact your Sitecore administrator to remove this form verification from the web form.

- Check User and Password – verifies the user name and password of the user in Sitecore. It uses the values in selected form fields to validate the Sitecore user's User Name, Password, and Domain. If verification fails, the message *The user name or password is incorrect* is displayed.
- Is User in Role – checks if the user is in the selected Sitecore role. This is often used together with the *Edit Role Membership* save action. This verification can fail if the user is not a member of the role or if the user is a member of the role. If verification fails, either *The user is not in the role* or *The user is in the role* message is displayed.
- User Exists – checks if the user has been created as a Sitecore user based on the values entered in the User Name and Domain fields. This is often used together with the *Create User* save action. This verification fails if the user does not exist or the user already exists. If verification fails, either *The user does not exist* or *The user already exists* message is displayed. You can also create new verifications in the Content Editor.

4. To configure a form verification, in the Added Form Verifications field, select the form verification that you want to configure and click Edit.
5. In the dialog box that appears, for example, the Check User and Password dialog box, fill in the relevant values and click OK.



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

Create a custom form verification action

A verification action is the first of three submit actions that are performed when a contact submits a form. When a form is submitted, form verifications check the values that have been entered in one or more fields in a form.

To indicate that a form verification has failed, the Web Forms for Marketers module throws an exception generated by the action code. The module handles the exception, and displays a custom error message to the visitor.

You can create custom form verification actions to suit your organization and its business needs.

To create a custom form verification action:

1. In Visual Studio, create a new project, for example `Sitecore.Forms.Sample`.
2. Add a new reference to the `Sitecore.Forms.Core` assembly.
3. Create a new class that inherits from the `Sitecore.Form.Core.Submit.BaseCheckAction` class.
4. When creating the new class, you can use the following code:

```
namespace Sitecore.Forms.Sample
{
  class SimpleVerification:BaseCheckAction
  {
    public override void Execute(Sitecore.Data.ID formid, IEnumerable<Form.Core.Controls.Data.ControlResult> fields)
    {
      foreach (ControlResult cr in fields)
      {
        PostedFile file= cr.Value as PostedFile;
        if(file!=null)
        {
          if (file.Data.Length > int.Parse(MaxSize))
          {
            throw new Exception("File size should be less than " + MaxSize+" bytes");
          }
        }
      }
    }
    public string MaxSize { get; set; }
  }
}
```

5. In the content tree, navigate to *sitecore/System/Modules/Web Forms for Marketers/Settings/Actions* and select the *Form Verification* item.
6. In the right pane, create a new item by clicking the Form Verification Action button. In the Message dialog, enter a name for the new item, and click OK.

7. In the new sample verification item that you created, on the right pane, in the Data section, fill in the Assembly and Class fields with the assembly and class names. Click Save.

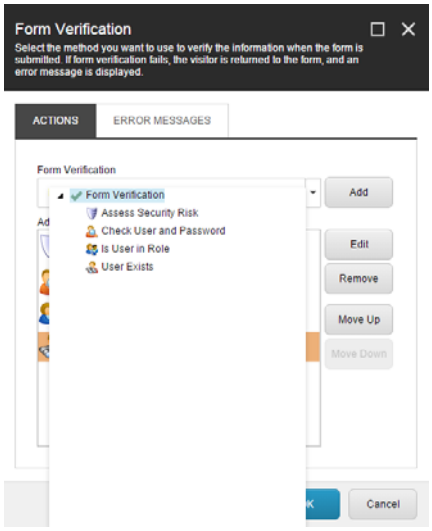
   The newly-created form verification item is now added to the list of default form verifications and is ready to be used.

Customize an error message for a submit action

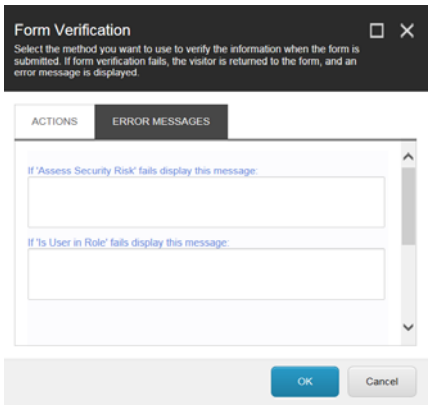
If a submit action fails, for example if a form verification where a user already exists or a save action fails because an email could not be sent, the Web Forms for Marketers module displays an error message. You can customize the error message to ensure that the visitor receives the best possible information.

To customize a submit action error message:

1. In the Form Designer, open a web form.
2. Click the form's Submit button.
3. To customize an error message for a form verification or save action, on the left pane, click Form Verification or Save Actions.
4. In the Form Verification or Save Actions dialog that appears, on the Actions tab, in the Form Verification / Save actions drop down list, click and select the action that you want to edit the error message for and then click Add.



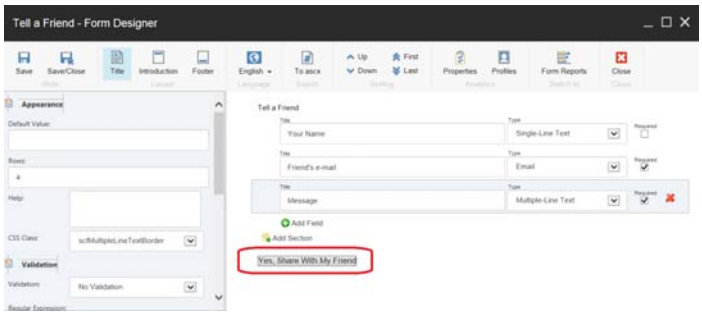
5. After you are finished with the selection, in the Form Verification or Save Actions dialog, click the Error Messages tab and enter the text of the new error messages for the actions you specified. Click OK.



If the submit action fails, the modified error message is displayed to the visitor.

Submit actions

To submit a web form, a website visitor must click the Submit button. The Submit button may have a different name than *Submit*, but it is always located under the last web form field section.

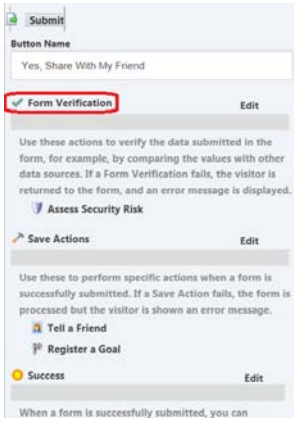


When a website visitor submits a web form, three types of actions are performed sequentially:

- Form verifications
- Save actions
- Success

Form verifications

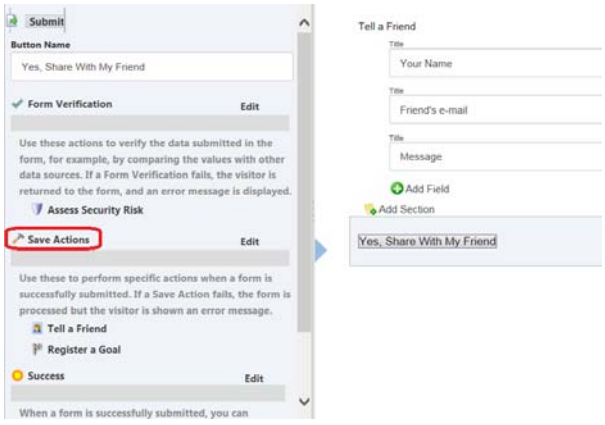
You can use web form verifications to check the values that have been entered in one or more fields in a form. When a user submits a web form, you can set up actions to verify the data submitted in the form.



For example, you can use web form verification to check a visitor's username and password or to verify the availability of a set of selected products against a product database. If form verification fails, the system returns the visitor to the web form, and the error message associated with the form verification is displayed. You can customize an error message for each individual form verification.

Save actions

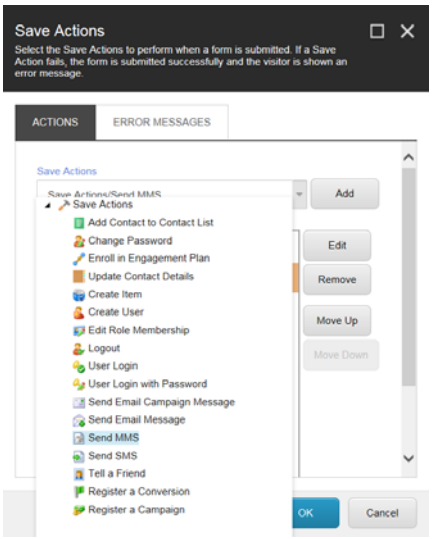
In the Web Forms for Marketers module, you can assign actions to a web form. They are performed when the visitor submits the form and all form verifications are successfully completed.



A save action is a Sitecore item and its purpose is to execute an action implemented in a .NET class. This class is specified in the Class and Assembly fields of the save action item. A save action item is based on the *Submit Action* template located in the folder */sitecore/templates/Web Forms for Marketers/Actions/Submit Action*.

Save action items are stored in the *Save actions* folder located in the folder *sitecore/ System/Modules/Web Forms for Marketers/Settings/Actions/Save Actions*). There are more than a dozen predefined (default) save actions that you can use.

If a save action fails, the visitor is not returned to the form.

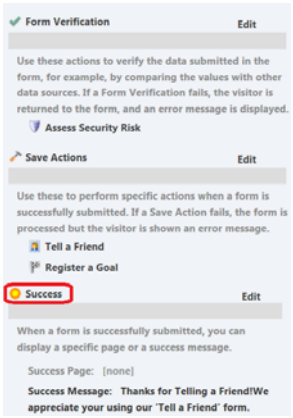


You can add, edit or delete save actions, create custom save actions and modify a save action using a field condition.

You can also configure save action fields.

Success

A success action is a submit action that lets you select either a Sitecore item or a message, which is presented to the visitor after they successfully submit a web form. This is the last of the three submit actions performed during a web form submission.



When the visitor successfully submits a form, the success message is displayed on the current page in place of the form. You can redirect the visitor to another page, using the Success Page field.

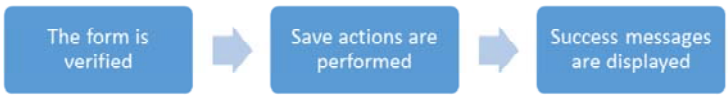
The pipeline that defines how a success action operates is called the `successAction` pipeline.

You can also add, edit or delete success messages.

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

Walkthrough: Configuring the Submit button

You can configure the Submit button in a variety of ways to give you more control over the information submitted by visitors through a web form. When a website visitor submits a web form, three types of actions are performed sequentially:



This walkthrough outlines how to:

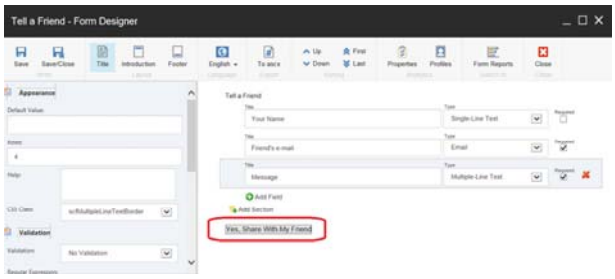
- Configure a form verification
- Specify a save action
- Specify a success action

To submit a web form, a website visitor must click the Submit button. The Submit button may have a different name than Submit, but it is always located under the last web form field section.

To configure the appearance of the Submit button, open the relevant form, and in the Form Designer, click the Submit button. In the left pane, you can specify the text on the button.

You can access the Form Designer through the:

- Content Editor
- Sitecore Desktop
- Experience Editor



Note

In addition to the default form verifications and save actions that are available, you can create custom form verifications, success messages, and save actions.

Configure a form verification

You can use web form verification to check the values that have been entered in one or more fields in a web form. When a user submits a web form, you can set up actions to verify the data submitted in the form.

You can configure the following default web form verifications:

- Assess Security Risk – verifies the information entered in form fields for content that may be malicious. If verification fails, the following message is displayed: The fields contain content that may present a security risk. Please enter appropriate information.

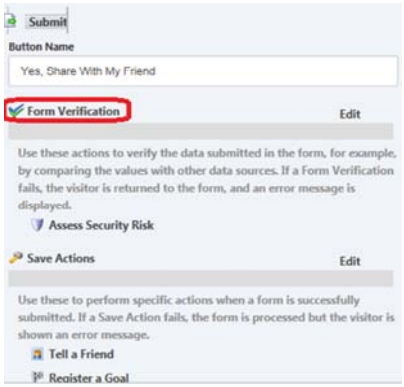
  Note

  This action is default for all web forms, and you cannot remove it. Contact your Sitecore administrator to remove this form verification from the web form.

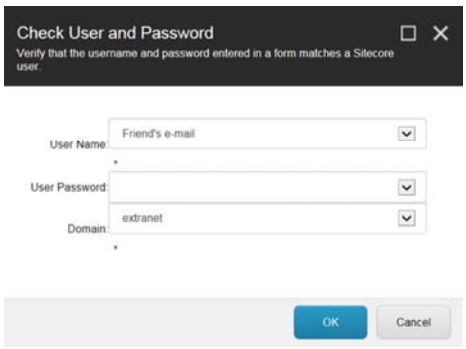
- Check User and Password – verifies the user name and password of the user in Sitecore. If verification fails, the following message is displayed: The user name or password is incorrect.
- Is User in Role – checks if the user is in the Sitecore role that you specified. You can specify that the form verification fails either if a user is a member of a particular role or if the user is not a member of a particular role. If verification fails, one of the following messages is displayed: The user is not in the role or The user is in the role.
- User Exists – checks if the user has been created as a Sitecore user based on the values entered in the User Name and Domain fields. You can specify that the form verification fails either if a user does not exist or if the user already exists in Sitecore. If verification fails, one of the following messages is displayed: The user does not exist or The user already exists.

  To configure a default web form verification:

1. In the Form Designer, in the left pane for the Submit button, click Form Verification.



2. In the Form Verification dialog box, in the Form Verification field, click the drop-down arrow and select the form verification that you want to add to the form, and then click Add.
3. To configure a form verification, in the Added Form Verifications field, select the form verification that you want to configure and click Edit.
4. In the dialog box that appears, for example, the Check User and Password dialog box, fill in the relevant values and click OK.



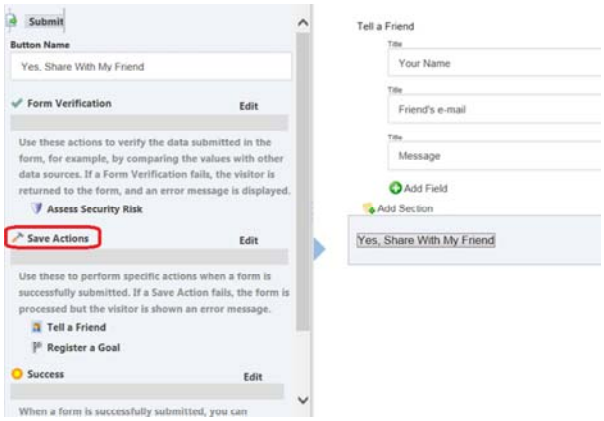
Specify a save action

You can assign certain save actions to a web form. A save action specifies what happens to the information in the web form after a visitor has successfully submitted it. There are several default save actions that you can use.

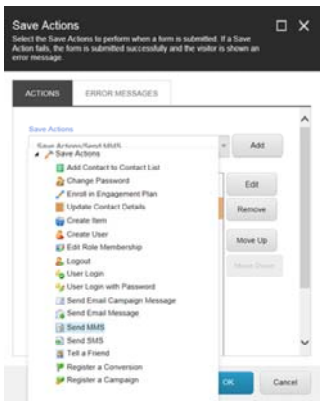
If a save action fails, the visitor is not returned to the form.

To specify a save action:

1. In the Form Designer, in the left pane for the Submit button, click Save Actions.



2. To add a save action, in the Save Actions dialog box, in the Save Actions field, click the drop-down arrow and select the relevant save action, and then click Add.



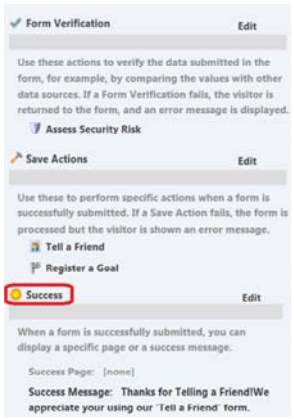
3. To delete an existing save action, in the Added Save Actions field, select the relevant save action and click Remove.
4. To configure a save action, in the Added Save Actions field, select the save action that you want to configure and click Edit.
5. In the dialog box that appears, for example, the Send MMS dialog box, fill in the relevant values and click OK.

Specify a success action

When a visitor successfully submits a web form, you can specify what happens, for example, a success message is displayed on the current page in place of the form. You can also redirect the visitor to another page, using the Success Page field. This is the last of the three submit actions performed during a web form submission.

To specify a success action:

1. In the Form Designer, in the left pane for the Submit button, click Success.



2. In the Success dialog box, select either:
   - Success Page – specify where you want to redirect the visitor to when the form is submitted successfully.

- Success Message – in the field, create a customized message that you want displayed when the form is submitted successfully.

3. Click OK.

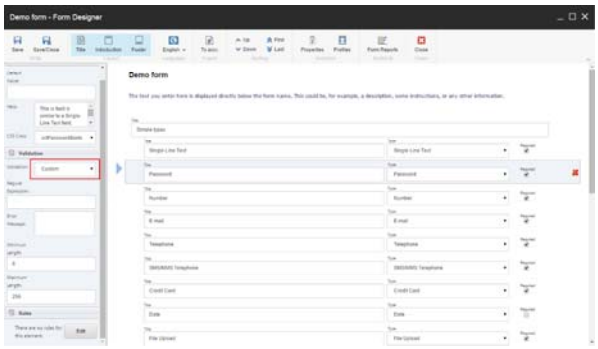
Create a custom form-specific field validation

A web form validation can help a website visitor enter a correct value in a field on your web form. By default, a custom, form-specific field validation is not configured and the Validation field in the Form Designer in the Validation section displays *No Validation*.

The rules that define a validation are displayed in the Regular Expression field, and they cannot be modified. However, if the validation is not found in the list of form-specific field validations, you can create a custom one that applies only to that field and form once selected.

To create a custom, form-specific field validation:

1. In the Form Designer, open the relevant web form and click the field that you want to assign a validation to, for example, *Password*.
2. On the left pane, in the Validation section, in the Validation field, click the drop-down arrow and select *Custom*.

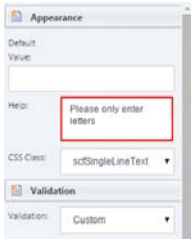


3. In the Regular Expression field, enter the expression you want to use as a validation.

   Note

   You can copy and paste a regular expression from another validation in this field and then modify it if necessary. For example, you can use the regular expression for *Letters Only* (`^[a-zA-Z]*$`), and then modify it so it specifies that a user has to enter a combination of upper and lowercase letters.

4. In the Help field, you can enter a description to tell users what type of information to enter.



5. In the Error Message field, you can specify the error message text that you want to appear when a website visitor enters incorrect text or an incorrect value in the field. For example: Please only use letters in this field.
6. Click Save or Save/Close to save your changes.

Create and assign a custom field-type validator

The Web Forms for Marketers module contains a number of field-type validations by default. When you assign validations to a field, the changes you have made to the field affect all the forms on your website that use it. In addition to the default validations, you can also create your own custom field-type validations.

You may want to create a custom field-type validator in case that default validators do not suit your needs. For example, you might want to ensure that all DatePicker fields on your website validate that the chosen date is in the future.

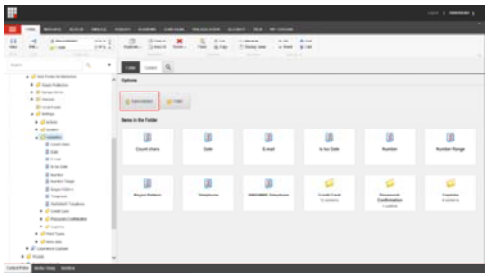
1. In Visual Studio, create a project.
2. In the *Sitecore.Forms.Core.dll* library, create a custom validation class, for example `Custom.Form.Validators.DateInFutureValidator` inherited from the `Sitecore.Form.Core.Validators.FormCustomValidator`.
3. Create a validation method of the *void* type that receives the object and `ServerValidateEventArgs` parameter types.

   See a code sample:

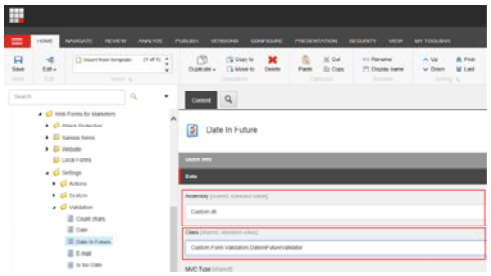
```
public class DateInFutureValidator : FormCustomValidator

{

 public DateInFutureValidator()

    {

      this.ServerValidate += this.OnDateValidate;
```

```
    }
    private void OnDateValidate(object source, ServerValidateEventArgs args)
    {
      DateTime time = Sitecore.DateUtil.IsoDateToDateTime(args.Value).Date;
      if (time >= DateTime.UtcNow)
      {
        args.IsValid = true;
        return;
      }
      args.IsValid = false;
    }
}
```

4. Compile your project to the */bin* folder.
5. In the content tree, navigate to the *sitecore/System/Modules/Web Forms for Marketers/Settings/Validation* folder and on the right pane, in the Folder tab, in the Options section, click on the BaseValidator button.



6. In the Message dialog box, enter a name for the new item, for example *Date In Future* and click OK. This creates a new item based on the *BaseValidator* template.
7. In the Data section of the item, in the Assembly and Class fields, enter the appropriate values of the custom assembly.

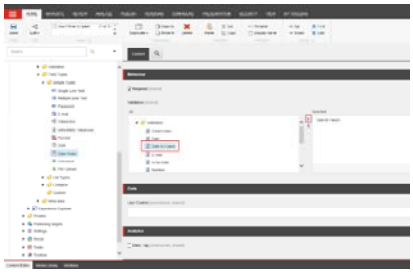


8. In the Error Message and Text fields, enter a relevant message for the visitor, for example, *The date is in the past, please select one in the future*.

   Note

   If you do not enter a message in the Text field, its value is copied from the Error Message field.

9. On the left pane, navigate to the relevant field type, for example Date Picker, located in the /sitecore/system/Modules/Web Forms for Marketers/Settings/Field Types/Simple Types/Date Picker folder.
10. On the right pane, in the Behavior section, in the Validation field, select the validator that you have created, for example, Date In Future and add it » to the Selected field.



Now that you have created the validator, you can assign it to a particular field type on your forms.

Implement the 'required' check box field validator

In the Web Forms for Marketers module, by default, the Check box field does not support the 'required' validation rule.

This topic outlines how you can make this field 'required' in the following ways:

- Use the Check box field and mark it as required
- Create a custom validator for the Check box field

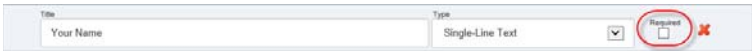
Note

This is only valid for web forms and does not support MVC forms.

Use the Check box field and mark it as required

To make the Check box field 'required' by using the field:

1. Add the Check box field to a web form.
2. Add one item to the list.
3. Mark this Check box as required.
4. Click Save.



Create a custom validator for the Check box field

To create a custom validator for the Check box field:

1. Create a class that is inherited from the `System.Web.UI.WebControls.BaseValidator` class. See the code sample:

```
class CheckboxValidation : BaseValidator
    {
        protected CheckBox ctrToValidate;
        protected CheckBox CheckBoxToValidate
        {
            get
            {
                if (ctrToValidate == null)
                {
                    ctrToValidate = base.FindControl(ControlToValidate) as CheckBox;
                }
                return ctrToValidate;
            }
        }
        protected override bool ControlPropertiesValid()
        {
            if (base.ControlToValidate == null || base.ControlToValidate.Length == 0)
            {
                throw new HttpException(string.Format("The ControlToValidate property of '{0}' cannot be blank.", this.ID));
            }
            if (this.CheckBoxToValidate == null)
            {
                throw new HttpException(string.Format("The CheckBoxValidator can only validate controls of type CheckBox."));
            }
            return true;
        }
        protected override bool EvaluateIsValid()
        {
this.ErrorMessage = string.Format(this.ErrorMessage, "{0}", CheckBoxToValidate.Text);
            //Validate whether checkbox is checked
            return this.CheckBoxToValidate.Checked == true;
        }
    }
```

2. In the content tree, navigate to the folder *sitecore/System/Modules/Web Forms for Marketers/Settings/Validation* and create an item by clicking the BaseValidator button.

   The item must be based on the *BaseValidator* template.

3. In the Message dialog, enter a name for the new item. Click OK.
4. In the right pane, in the Assembly and Class fields, enter the relevant values from your custom assembly.
5. In the Validator section, in the Error Message field, enter the following string: `The {0} check box must be checked.`
6. In the Text field, enter a relevant message. If this field is blank, its value is the same as the one in the Error Message field.
7. Duplicate the check box item (*/sitecore/System/Modules/Web Forms for Marketers/Settings/Field Types/Simple Types/checkbox*) and rename it to *CheckBoxRequired*.

Note

In the *CheckBoxRequired* item, do not select the *Required* check box.

8. In the *CheckBoxRequired* item (*/sitecore/system/modules/web forms for marketers/settings/field types/simple types/CheckBoxRequired*), in the Validation field, select your custom validator.

Set up and create a form-specific field validation

In addition to field-type validations, the Web Form for Marketers module includes form-specific field validations that are an extended set of validations applied directly on a field of a specific form.

Using a form-specific field validation makes it simple to add the most frequently used types of validations to fields in your new or existing web forms. You can also add a new, form-specific field validator to the list of existing validations, making it easier and faster to select it next time you want to assign a validation to a form.

For example, in a field where you want users to only enter numbers, you can select the *Numbers Only* validation to display an error message if they enter symbols or letters.

You can find the list of default field validators in the content tree in the *Predefined Validators* folder *(/sitecore/system/Modules/Web Forms for Marketers/Settings/Meta data/Predefined Validators)*

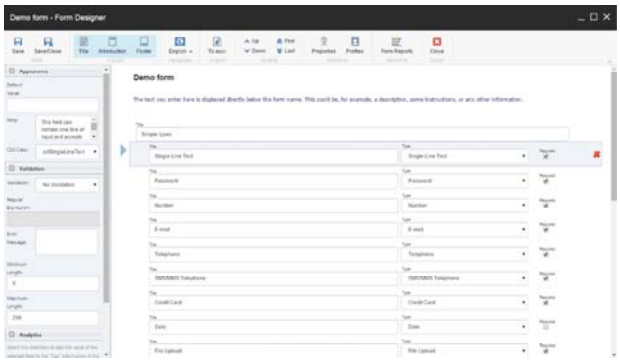
The rules that define a validation are displayed in the Regular Expression field, and they are fixed. However, if you create a custom form-specific field validation, you can manually configure the regular expression.

This topic outlines the following procedures:

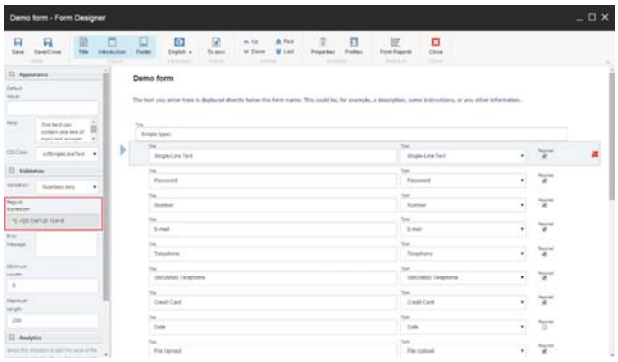
- Set up a form-specific field validation
- Create a new, form-specific field validator

Set up a form-specific field validation

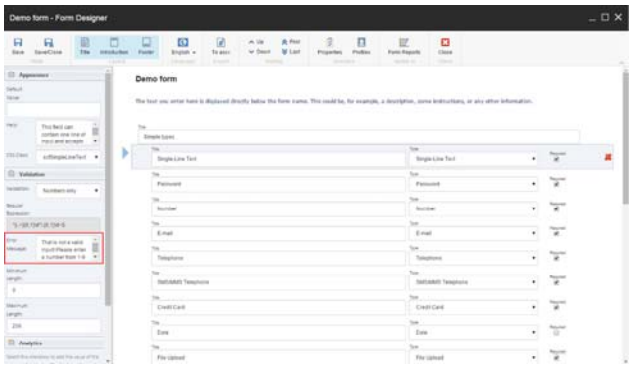
1. Open the Form Designer, and in the Select a Form dialog box, choose the form that you want to set up, and click OK.
2. In the Form Designer, click the relevant field where you want to add the validator, for example, *Single-Line text.*



3. In the left pane, in the Validation section, click the drop-down arrow in the Validation field and select the relevant validator from the list, for example, *Numbers only*. The Regular Expression field is filled in automatically.



4. In the Error Message field, you can customize the message that appears if a user enters the wrong values.



5. Specify the minimum and maximum length of the numbers of characters allowed in the form. If the number of characters entered in the field on the web form is greater than the number specified in the Maximum Length field, an error message appears.

   Note

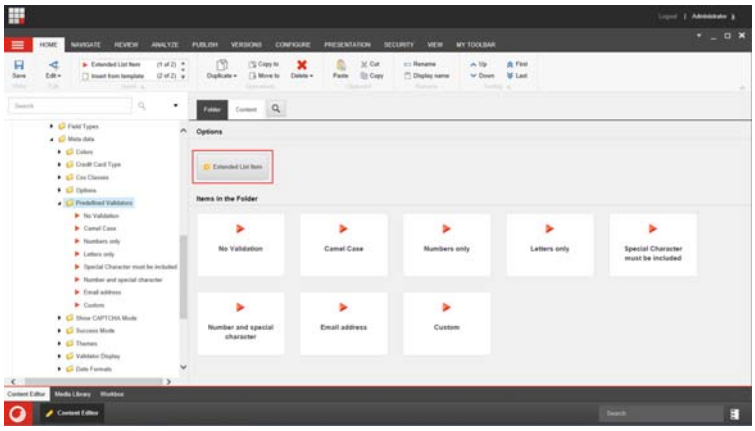
   Each field can have a set of validations. On the left pane, under the Validation section users can also choose a custom validation. These settings work in combination with validations assigned to the field type.

6. Click Save.

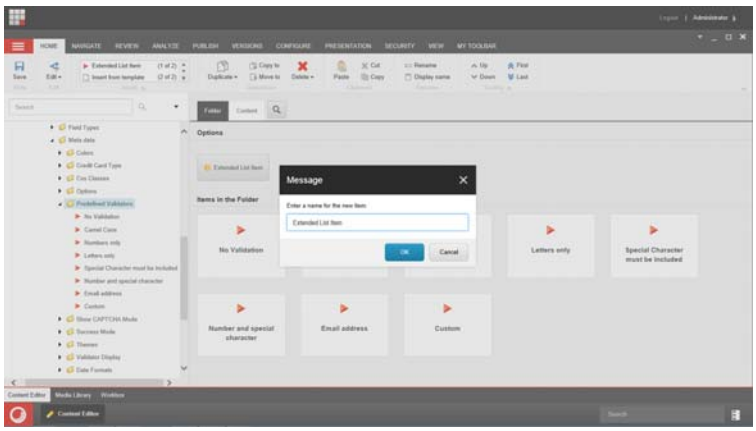
Create a new form-specific field validator

To create a new, predefined form-specific field validator to the list, you must add an item based on the *Extended List Item* template in the *Predefined Validators* folder:

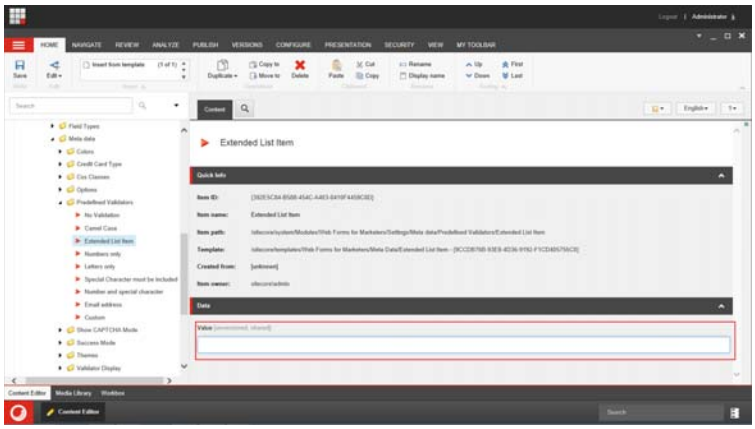
1. On the Sitecore Desktop, open the Content Editor, and navigate to the *Predefined Validators* folder (`sitecore/System/Modules/Web Forms for Marketers/Settings/Meta data/Predefined Validators`).
2. On the Folder tab, in the Options section click Extended List Item. Alternatively, in the ribbon, on the Home tab, in the Insert group, click Extended List Item.



3. In the Message dialog box, enter a name for the new item and click OK.



4. Click the Content tab, and in the Data section, in the Value field, fill in the appropriate value in the format of a regular expression. For example, for a *Numbers only* validation: `^[-,+]{0,1}\d*\.{0,1}\d+$`



5. Click Save to save your changes.

   The new form-specific field validator is now displayed in the list of validations in the Form Designer.

The field-type validations

A good web form usually contains one or more validations for one or more of its fields. A validation is a programmatically hard-coded condition that specifies whether the text you enter in a field of a web form can be accepted as a valid value. If the text entered does not conform to the predefined format, an error message is displayed in the web form field.

In the Web Forms for Marketers module, validations on a field type apply to all forms that use fields of that type (for example, Single-Line text) and these change if you change the field type, whereas form-specific validations stay with the field regardless of field-type.

The module supports web forms (.aspx) and MVC (.cshtml) layouts. Under field-type validations, both web forms and MVC are implemented in different ways. For web forms, a field type validation consists of a code and items that are stored in the content tree, in the validation item *sitecore/system/modules/Web Forms for Marketers/settings*. For MVC, a field type validation consists only of a code and does not use the validation item structure. Each field type contains predefined built-in validations relevant for the particular field type, for example: email, phone number, or credit card validations.

You can add a new, custom field type validation or custom form-specific field validation to a form, you can assign more than one validation to a field type, or you can remove them according to your needs.

Field-type validations for web forms:

Validation	Description	Used by field type
Count chars	Checks the number of characters in a string. You can set the minimum and maximum number of characters.	Single-Line Text Multiple-Line Text Password Password-Confirmation
Date	Checks whether the value entered is a date.	Date
Email	Checks whether the value entered uses the format of an email address.	Email
Is Iso Date	Checks whether the value entered is an ISO date.	
Number	Checks whether the values entered are numbers (negative numbers and integers are allowed).	Number
Number range	Checks whether the values entered are within a specified range of numbers.	Number
Regex pattern	Checks whether the values entered conform to a rule that you specify.	Single-Line Text Multiple-Line Text Password Password-Confirmation
Telephone		Telephone

	Checks whether the value entered is a valid telephone number.	
SMS/MMS Telephone	Checks whether the value entered is a valid format for a text message or an MMS.	SMS/MMS Telephone
Luhn formula		
American Express, Diners Club Carte Blanche, Diners Club International, Diners Club US and Canada, JCB, Maestro, MasterCard, Solo, Switch, Visa, Visa Electron	Checks the validity of a credit card number based on the type of credit card.	Credit card
Compare Password - Confirmation	Compares the values entered in the Password and Confirmation fields.	Password-Confirmation
Captcha	Compares the text displayed in a CAPTCHA field with the value entered by the user.	Captcha
Robot Protection	Checks whether a current visitor is a robot (using the robot detection algorithm).	Captcha
Suspicious Visitor	Identifies a website visitor who submits a web form several times during a short period of time as suspicious.	Captcha
Suspicious Form Activity	Identifies the form activity as suspicious if a web form is submitted several times during a short period of time by one or many users.	Captcha

Field-type validation attributes for MVC:

Description	Model name	
Checks the validity of a credit card number based on the type of credit card.	CreditCardField	`[CreditCard]` `public override string Value { get; set; }`
Checks whether or not the value entered is a date.	DateField	`[ParameterName("SelectedDate")]` `public override string Value`

Checks whether or not the value entered uses the format of an email address.	EmailField	`[DynamicEmail("EmailRegularExpression")]`
		`[DataType(DataType.EmailAddress)]`
		`public override string Value { get; set; }`

Checks whether or not the values entered are numbers (negative numbers and integers are allowed). Also checks whether or not the values entered are within a specified range of numbers.	NumberField	`[RegularExpression(@"^[-,+]{0,1}\d*\.{0,1}\d+$", ErrorMessage = @"Field contains an invalid number.'`
		`[DynamicRange("MinimumValue", "MaximumValue", ErrorMessage = @"The number in {0} must be at least {1`
		`public override string Value { get; set; }`

Compares the values entered in the Password and Confirmation fields.	PasswordConfirmationField	`[DataType(DataType.Password)]`
		`[DynamicCompare("Value", "PasswordTitle", "ConfirmationTitle", ErrorMessage = @"The {0} and {1} fiel`
		`public string Confirmation { get; set; }`

Compares the text displayed on an image or sound voiced by a speaker with the value entered by the user.	RecaptchaField	`[RequestFormValue("g-recaptcha-response")]`
		`[RecaptchaResponseValidator(ParameterName = "RecaptchaValidatorError")]`
		`public override string Value { get; set; }`

Checks whether or not the text in a field is within the specified min and max length.	SingleLineTextField	`[DataType(DataType.Text)]`
		`[DynamicStringLength("MinLength", "MaxLength", ErrorMessage = @"The field {0} must be a string with`
		`public override string Value { get; set; }`

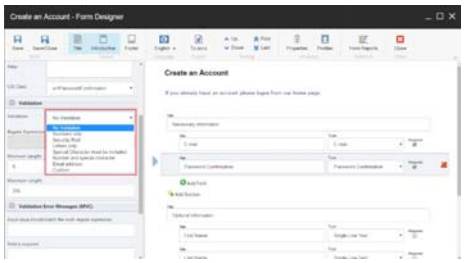
Checks whether or not the text is an SMS number	SmsTelephoneField	`[DynamicRegularExpression(@"^\+?\d{3,}", null, ErrorMessage = @"{0} contains an invalid sms/mms tele` `public override string Value { get; set; }`
Checks whether or not the text is a phone number	TelephoneField	`[DataType(DataType.PhoneNumber)]  [DynamicRegularExpression(@"^\+?\s{0,}\d{0,}\s{0,}(\(\s{0,}\d{1,` `public override string Value { get; set; }`

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

The form-specific validations

Validations are programmatically hard-coded conditions that specify whether the information entered in a field of a web form is a valid value. The Web Forms for Marketers module contains two kinds of form validations, form-specific and field type, but you can also create custom ones.

Form-specific validations stay with the field regardless of the type of field and make it easy to add the most frequently used types of validations to the fields in your form. The rules governing the validation are displayed in the Regular Expression field. These regular expressions are fixed and cannot be modified unless you choose the *Custom* validation.



Validation	Description	Default Regular Expression
Numbers only	Checks whether the values entered are numbers.	^[-,+]{0,1}\d*\.{0,1}\d+$
Security Risk	Checks the field input for content that may be malicious, for example code script.	^((?!([<&][\S])).)*$
Letters only	Checks whether the values entered are letters.	^[a-zA-Z]*$
Special Character must not be included	Checks whether the field contains only letters and whitespaces.	^[a-zA-Z\s]*$

Number and word characters	Checks whether the field contains only numbers, letters or whitespaces.	^[\w\s]*$
Email address	Checks whether the value entered uses the format of an email address.	^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}$
Custom	Custom	You can enter your own regular expression.

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