



Sitecore E-Commerce Services 2.0 on CMS 6.6

Quick Start Guide

How to build a basic webshop

Table of Contents

Chapter 1	Introduction	3
1.1	Building a Webshop	4
1.2	SES Documentation	6
Chapter 2	Product Repositories	8
2.1	Creating the Product Repositories Root Folder	9
2.2	Configuring the Website Backend Search.....	11
2.2.1	Comparison of Sitecore Query, Lucene Search, and Fast Query.....	11
2.2.2	Setting up Lucene Search.....	12
	Publishing the Entire Website.....	12
	Setting up Product Search.....	12
	Rebuilding the Search Indexes.....	14
	Rebuilding the Link Database.....	14
2.3	Creating Custom Product Templates	15
2.3.1	Configure the Specification Section	18
2.4	Creating Products and Product Categories in a Product Repository	19
2.4.1	The Optimal Webshop Repository Structure	19
2.4.2	Product Variant and Master Product	21
2.4.3	Creating Product Categories in a Product Repository.....	22
2.4.4	Creating Products Items in a Product Repository	23
	Extending the Product Template	23
	Configuring Product Prices.....	23
2.5	Creating the Root Folder for your Webshop.....	24
Chapter 3	Configuring Settings	26
3.1	Webshop Site Settings	27
3.1.1	Creating Webshop Site Settings Item.....	27
3.2	Business Catalog Settings.....	29
3.3	General Settings	30
Chapter 4	Website.....	32
4.1	Registering Your Website.....	33
4.1.1	Order Manager Site Context Name.....	34
4.2	Building the Webshop Pages.....	35
4.2.1	Creating the Product Category Pages.....	35
	Creating a Product Category Page	35
	Creating a Product Category Folder	36
4.2.2	Adding Repository Products to Webshop Pages	37
4.3	Configuring the Frontend Search on the Webshop.....	40
Chapter 5	Product Presentation	43
5.1	Configuring the Product Catalog.....	44
	Product Catalog Configuration Prerequisites	44
	Creating a Search Options List.....	45
5.2	Configuring the Presentation Settings.....	50
5.2.1	Product Details and Presentation.....	50
	Creating the Product Details Presentation	53
5.2.2	Product Display Modes.....	55
5.2.3	Product Selection Method.....	56
Chapter 6	Business Settings.....	57
6.1	Webshop Business Settings	58
6.1.1	Overview of Webshop Business Settings.....	58
6.1.2	Creating the Webshop Business Settings Item.....	59
6.1.3	Selecting the Right Webshop Business Settings	60
6.1.4	Creating a Webshop Business Setting Option.....	62
6.1.5	Payment Options.....	63
6.2	Creating Checkout Process	65

Chapter 1

Introduction

This document describes how to build a webshop using only the Sitecore E-Commerce Services (SES) core package. It is useful for website administrators and developers. The core package does not contain the example pages.

This document contains the following chapters:

- **Chapter 1 — Introduction**
This chapter contains a brief description of this manual.
- **Chapter 2 — Product Repositories**
This chapter describes how to build a webshop from the SES core package.
- **Chapter 3 — Configuring Settings**
This chapter describes how to configure the site, business catalog, and general settings.
- **Chapter 4 — Website**
This chapter describes how to register, build, and configure the website.
- **Chapter 5 — Product Presentation**
This chapter describes how to configure the product catalog.
- **Chapter 6 — Business Settings**
This chapter describes how to configure the webshop business settings.

1.1 Building a Webshop

If you decide to build a webshop using the core package, you must build it in the following order:

1. Products and Product Repositories
 - a. Create the *Product Repositories* root folder
 - b. Configure website backend search
 - i. Compare Sitecore Query, Lucene Search and Fast Query
 - ii. Set up Lucene search
 - c. Create custom product templates
 - i. Configure the Specification section
 - d. Create products and product categories in a product repository
 - i. Learn about the optimal webshop repository structure
 - ii. Learn about the difference between the product variant and master product
 - iii. Create product categories in a product repository
 - iv. Create products items in a product repository
 - e. Create the root folder for your webshop
 - f. Create the home item for your webshop
2. Configuring the Site Settings
 - a. Configure Webshop Site Settings
 - i. Create the Webshop Site Settings item
 - b. Configure Business Catalog Settings
 - c. Configure General Settings
3. Website
 - a. Register your website
 - b. Build your webshop pages
 - i. Create product category pages
 - ii. Add repository products to webshop pages
 - c. Configure the Website frontend search
4. Product Presentation
 - a. Configure the Order Catalog and the Product Catalog
 - i. Order Catalog / Product Catalog configuration prerequisites
 - ii. Create a search options list
 - iii. Configure search options in an existing checkbox list
 - b. Configure presentation settings
 - i. Configure product details and presentation
 - ii. Create product details presentation
 - iii. Configure product display modes
 - iv. Configure product selection method

5. Business Settings

- a. Learn about different types of Webshop Business Settings
- b. Create the Webshop Business Settings item
- c. Select the right webshop business settings
- d. Create a Webshop Business Setting option
- e. Create the checkout process

1.2 SES Documentation

For more information about SES, see the following documents:

- **The SES Installation Guide**

SES is a framework for implementing e-commerce solutions on the Sitecore platform. This document describes how to install SES, the SES Example pages, and Order Manager. It also describes how to configure search and some more advanced topics, such as, distributed environments and scalability.
- **The SES Configuration Guide**

This guide lists and describes the settings that you must configure in the Content Editor when you implement an e-commerce solution. SES comes with a prototype webshop and most of the functionality is described in terms of how it is implemented in this sample site.
- **Sitecore E-Commerce Services MvpWebStore**

This document describes the MvpWebStore project. This is a `WebForms` based project that demonstrates the latest features of Sitecore E-Commerce Services version 2.0. You can download this project and its documentation from the [Marketplace](#).
- **The Sitecore E-Commerce Cookbook**

This user guide describes how marketers and webshop managers can manage their product catalog and organize the way that products are displayed in their webshop. The topics covered include managing your product catalog, creating and editing product categories, specifying how the products should be displayed in the webshop, managing campaigns, and analyzing the traffic on the webshop.
- **The Sitecore E-Commerce DMS Cookbook**

This manual describes how marketers and webshop managers can utilize the Sitecore Digital Marketing System (DMS) in combination with E-Commerce Services to analyze the shopping behavior of customers and to assess the overall effectiveness of their e-commerce websites.
- **The Sitecore Order Manager Cookbook**

This guide explains how you can use the Order Manager to process and manage orders. The content in this guide includes a tour of the Order Manager UI and several walkthroughs that provide step by step instructions on how to complete typical Order Manager scenarios.
- **The SES Developer's Cookbook**

This document helps Sitecore administrators and developers understand, customize, and extend the functionality of SES.
- **The Sitecore E-Commerce API Reference Guide**

This guide describes the SES API and some useful extensions to its functionality. It describes the contract/class functionality, parent classes, implementation, important methods/properties and sample code examples.
- **The SES Payment Provider Guide**

This is a brief guide to the payment methods supported by SES. It also contains instructions on how to build a new custom payment provider using Sitecore PayPal implementation as an example. The document also includes information about the payment provider API.
- **The Order Manager API Reference Guide**

This guide describes the Order Manager API and some useful extensions to its functionality. It describes the class functionality, parent classes, implementation, methods, and properties.

- **The Order Manager Developer Cookbook**

This guide describes how to customize and extend the Order Manager application.

You can download all of these documents — except the MvpWebStore documentation — from the [Sitecore Developer Network](#).

Chapter 2

Product Repositories

This chapter describes how to configure the product repositories of your webshop, configure the search of your webshop, create custom product templates, create products and product categories in a product repository. Moreover, you will know how to create an optimal webshop repository structure, the difference between the product variant and the master product. Finally, you will know why it is important to create the root folder and the home item for your webshop.

The chapter contains the following sections:

- Creating the Product Repositories Root Folder
- Configuring the Website Backend Search
- Creating Custom Product Templates
- Creating Products and Product Categories in a Product Repository
- Creating the Root Folder for your Webshop

2.1 Creating the Product Repositories Root Folder

Before you create the product repositories, you must create the *Product Repositories* item to store your product repository or multiple product repositories. If you use Lucene search for indexing, it is more efficient that you create all the product repositories in one place. This is also applicable if you want to create more webshops in the future.

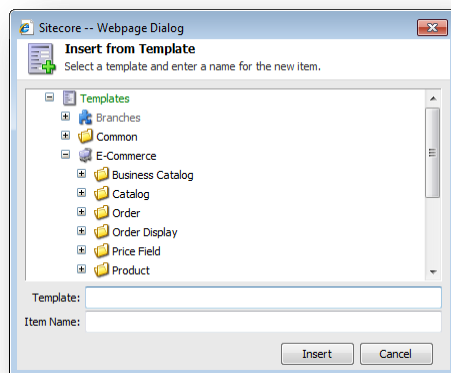
Note

If you want to create several webshops, you might need several repositories under the Product Repositories item.

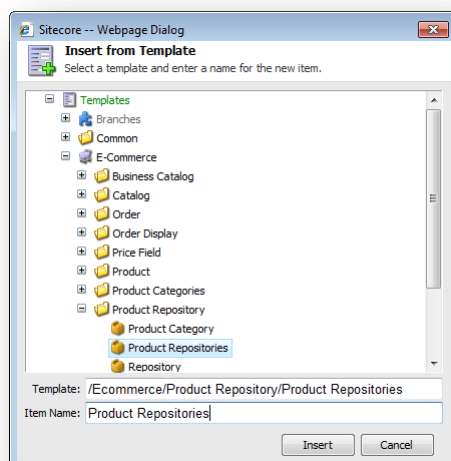
We recommend that you create the *Product Repositories* item under *Sitecore/Content*.

To create a *Product Repositories* item:

1. Right-click the *Sitecore/Content* item and click the **Insert from Template** option. The **Insert from Template** dialog box appears.



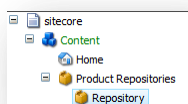
2. Navigate to the *Templates/E-Commerce/Product Repository/Product Repositories* template and call it *Product Repositories*.



3. Click **Insert**, and the *Product Repositories* item appears under the Sitecore/Content item.



4. To add a product repository, right-click the *Product Repositories* item and, in the menu that opens, click the **Repository** option. Give it a name and click **OK**.



After you have added the product repositories, you must indicate the path to the *Product Repositories* item in your search configuration settings to make the search for products possible.

Important

If you use Lucene search in Sitecore, you can add one *Product Repositories* folder to your website. SES can therefore refer to one Lucene index. Each webshop must refer to its own frontend Lucene index but can use same index for product repository. Therefore, every webshop repository must be stored under one item, in this case the *Product Repositories* item. If you use Sitecore Query or Fast Query, you do not have this constraint. For more information about the differences between Lucene search, Sitecore Query, and Fast Query, see the chapter *Comparison of Sitecore Query, Lucene Search, and Fast Query*.

2.2 Configuring the Website Backend Search

SES uses search providers to search for products using the API both on the website and in the Content Editor.

By default, SES supports three different search providers:

- Sitecore Query
- Fast Query
- Lucene Search

You can configure other search providers in SES if you want to.

2.2.1 Comparison of Sitecore Query, Lucene Search, and Fast Query

Before you decide which search providers to use for the Product Catalog and Order Catalog in your webshop, you must know the differences between Sitecore Query, Lucene Search, and Fast Query.

Comparison Criteria	Sitecore Query	Lucene Search	Fast Query
Performance	Normal	Higher	Highest
Is case sensitive?	Yes	No	No
Supports standard values?	Yes	Yes	No
Supports functions?	Yes	No	No
Supports sub queries?	Yes	No	Partially
Requires time to rebuild indexes after update\create	No	Yes	No
Supports range queries search?	Yes	No Note The search for the price range [5 — 100] is currently not supported.	Yes

Tip

We recommend that you use Lucene search for the Product Catalog because of its high performance and support for standard values. Use Fast Query for orders in which standard values are irrelevant.

For more information about Sitecore Query, see the [Using Sitecore Query article](#).

For more information about Sitecore Fast Query, see the [Using Sitecore Fast Query manual](#).

For more information about Lucene Search, see the [Sitecore Search and Indexing document](#).

For more information about the differences between the 3 search engines, see this [Sitecore blog](#) about options to query items in Sitecore.

2.2.2 Setting up Lucene Search

Note

The procedure described in this section only applies to Lucene search.

Lucene search can be used for resolving products so that they can be presented on the website.

To configure the search indexes for your SES installation:

1. Publish the entire website.
2. Set up product search (or Product catalog search) by implementing changes to the `.config` files.
3. Rebuild the search indexes.
4. Rebuild the link database.

Important

You must perform these tasks in this order.

Publishing the Entire Website

You publish the entire website to make the `index ID` in the `search/configurations/indexes` section visible for the crawler, a computer program that is capable of performing recursive searches on the World Wide Web.

```
<search>
  <configuration>
    <indexes>
      <index id="products" type="Sitecore.Search.Index, Sitecore.Kernel">
        <param desc="name">$(id)</param>
        <param desc="folder">__products</param>
        <Analyzer type="Sitecore.Ecommerce.Search.LuceneAnalyzer, Sitecore.Ecommerce.Kernel" />
        <!--<locations hint="list:AddCrawler"><master type="Sitecore.Ecommerce.Search.DatabaseCrawler,
      </index>
    </indexes>
  </configuration>
</search>
```

Setting up Product Search

Before you can use the product search forms to search for products in Sitecore, you must ensure that the `Sitecore.Ecommerce.config` file points to the product repository where you store your product items in Sitecore.

The `Sitecore.Ecommerce.config` file is stored in the `Website\App_Config\include` folder.

To make the search point to the product repository:

1. In the `Sitecore.Ecommerce.config` file, navigate to the `indexes` section.

```
<search>
  <configuration>
    <indexes>
      <index id="products" type="Sitecore.Search.Index, Sitecore.Kernel">
        <param desc="name">$(id)</param>
        <param desc="folder">__products</param>
        <Analyzer type="Sitecore.Ecommerce.Search.LuceneAnalyzer, Sitecore.Ecommerce.Kernel" />
        <!--<locations hint="list:AddCrawler"><master type="Sitecore.Ecommerce.Search.DatabaseCrawler,
      </index>
    </indexes>
  </configuration>
</search>
```

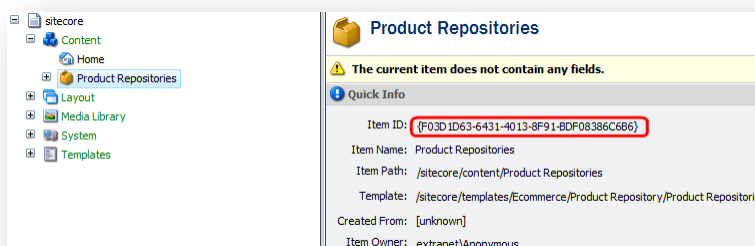
- Uncomment the section in green and insert it after `<Analyzer type="Sitecore.Ecommerce.Search.LuceneAnalyzer, Sitecore.Ecommerce.Kernel" />`, so that it appears in the code in the following way:

```
<search>
  <configuration>
    <indexes>
      <index id="products" type="Sitecore.Search.Index, Sitecore.Kernel">
        <param desc="name">$(id)</param>
        <param desc="folder">__products</param>
        <Analyzer type="Sitecore.Ecommerce.Search.LuceneAnalyzer, Sitecore.Ecommerce.Kernel" />
        <locations hint="list:AddCrawler">
          <master type="Sitecore.Ecommerce.Search.DatabaseCrawler, Sitecore.Ecommerce.Kernel">
            <Database>master</Database>
            <Root>{F03D1D63-6431-4013-8F91-BDF08386C6B6}</Root>
            <Tags>master products</Tags>
          </master>
          <web type="Sitecore.Ecommerce.Search.DatabaseCrawler, Sitecore.Ecommerce.Kernel">
            <Database>web</Database>
            <Root>{F03D1D63-6431-4013-8F91-BDF08386C6B6}</Root>
            <Tags>web products</Tags>
          </web>
        </locations>
      </index>
    </indexes>
  </configuration>
</search>
```

- In the inserted `locations` section, replace both of the `<Root>{.....}</Root>` GUIDs with the Item ID or GUID of the product repository where you store your products.

```
<search>
  <configuration>
    <indexes>
      <index id="products" type="Sitecore.Search.Index, Sitecore.Kernel">
        <param desc="name">$(id)</param>
        <param desc="folder">__products</param>
        <Analyzer type="Sitecore.Ecommerce.Search.LuceneAnalyzer, Sitecore.Ecommerce.Kernel" />
        <locations hint="list:AddCrawler">
          <master type="Sitecore.Ecommerce.Search.DatabaseCrawler, Sitecore.Ecommerce.Kernel">
            <Database>master</Database>
            <Root>{F03D1D63-6431-4013-8F91-BDF08386C6B6}</Root>
            <Tags>master products</Tags>
          </master>
          <web type="Sitecore.Ecommerce.Search.DatabaseCrawler, Sitecore.Ecommerce.Kernel">
            <Database>web</Database>
            <Root>{F03D1D63-6431-4013-8F91-BDF08386C6B6}</Root>
            <Tags>web products</Tags>
          </web>
        </locations>
        <!--<locations hint="list:AddCrawler"><master type="Sitecore.Ecommerce.Search.DatabaseCrawler,
        </index>
      </indexes>
    </configuration>
  </search>
```

The item ID of both of the root items must be identical with the item ID of the *Product Repositories* item.



The screenshot shows the Sitecore Content Editor interface. On the left, a tree view shows the site structure with 'Product Repositories' selected. On the right, the 'Product Repositories' item details are displayed. A warning message states 'The current item does not contain any fields.' Below this, the 'Quick Info' section shows the following details:

- Item ID: {F03D1D63-6431-4013-8F91-BDF08386C6B6}
- Item Name: Product Repositories
- Item Path: /sitecore/content/Product Repositories
- Template: /sitecore/templates/Ecommerce/Product Repository/Product Repositories
- Created From: [unknown]
- Item Owner: extranet\Anonymous

When you rebuild the search indexes, a new index called *products* is added.

Rebuilding the Search Indexes

After you have implemented these changes in the appropriate config files, you must build the search indexes.

To rebuild the search index:

1. Open the **Sitecore Desktop**.
2. Click **Sitecore, Control Panel, Database, Rebuild the Search Index**.
3. In the wizard, select all of the search indexes and click **Rebuild**.

Rebuilding the Link Database

After you have implemented these changes in the appropriate config files, you must build the link database.

You must rebuild the link database to resolve the settings, for example the settings for products, if they are not located under the SES website. If you store all the products and other settings under the SES website, you don't have to rebuild the link database. However, we recommend that you perform this action to make sure that all the links are updated.

To rebuild the search index:

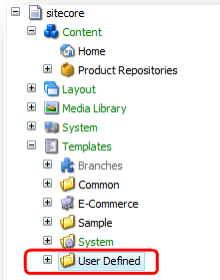
1. Open the **Sitecore Desktop**.
2. Click **Sitecore, Control Panel, Database**, and then click **Rebuild the Link Database**.
3. In the wizard, select all of the link databases and click **Rebuild**.

2.3 Creating Custom Product Templates

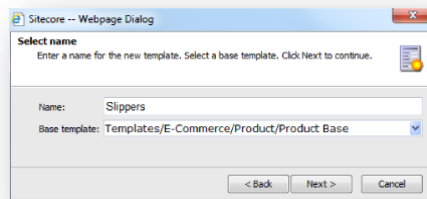
Although the E-Commerce core package already contains a *Product* template, you may need to create other templates for products with other sets of properties or specifications. You can either extend the existing *Product* template or create a custom product template.

To create a custom product template:

1. Navigate to the `Content/Templates/User Defined` folder.



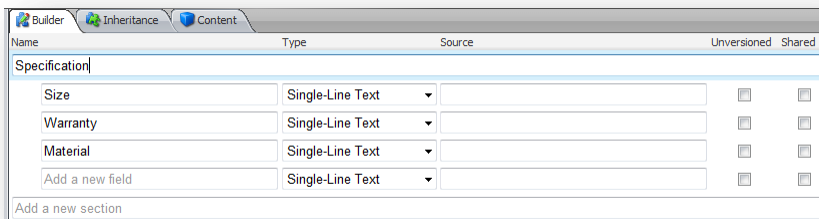
2. In the right-hand pane, on the **Folder** tab, click the **New Template** item.



3. Fill in the **Name** and the **Base Template** fields.

In the **Base Template** field, select the *Product Base* template from the `Templates/Ecommerce/Product` folder.

4. On the **Builder** tab, fill in the fields that you want to define the appearance of the current product group template.

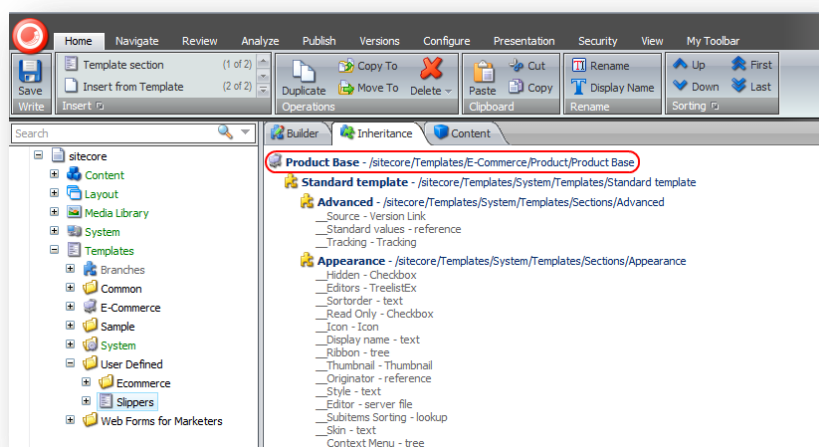


Important

Add a new section and call it *Specification*. If the product specifications are stored in a field section with a different name, you must create custom code classes to map to the product templates so that the script can read the product data using the API. The template section called *Specification* has a special meaning for the script. All its fields are mapped to a special dictionary collection on the product code class that maps to the product template.

For more information about the *Specification* field section, see the *Sitecore E-Commerce API Reference Guide*.

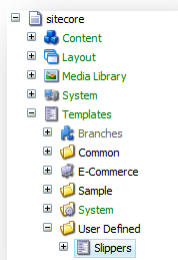
- On the **Inheritance** tab, you can see which templates this template is based on.



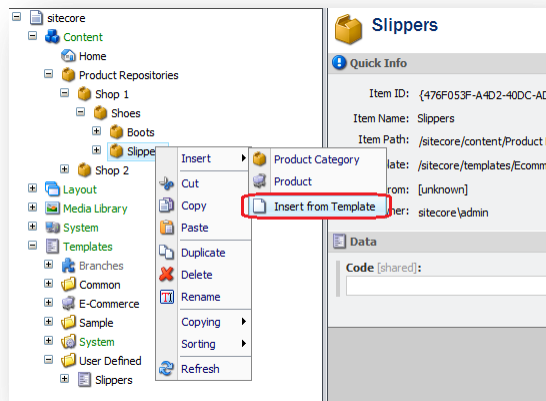
Tip

You can use inheritance to build different product types or product variants. In other words, you do not have to create a new product template from scratch. You can re-use the existing product template and extend it by adding one or more new field sections or by changing a few fields.

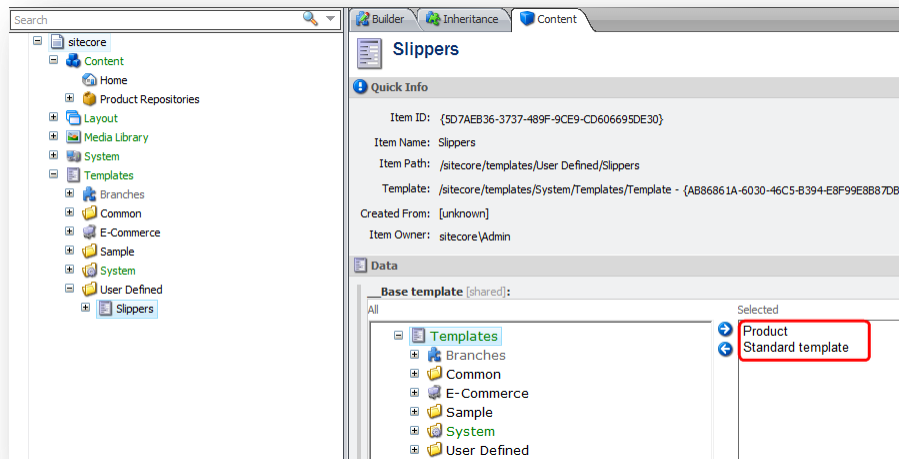
- On the **Content** tab in the **_Base Template** field, you can add more templates to the **Inheritance** tab.
- When you are finished, save your changes.



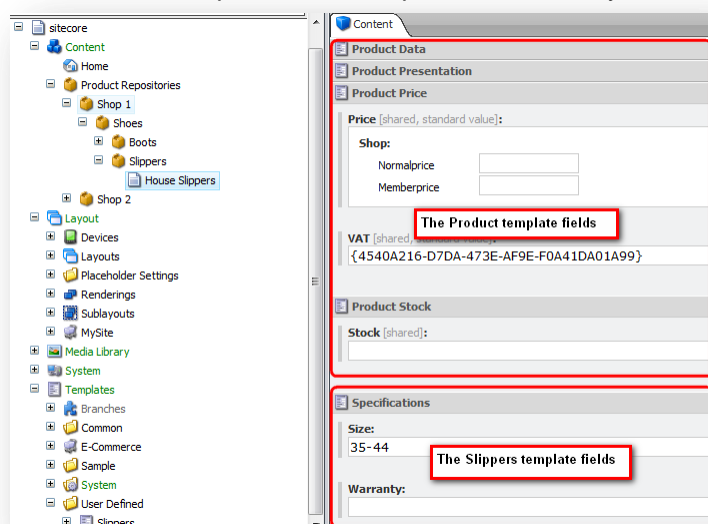
You can use this template to create products. A product that you create with this template has the section and fields that you added.



- Let us assume that you inserted the following templates on the **Content** tab of the *Slippers* template and in the following order.



The order of the template fields in a product item that you create will be the same.



For more information about how to create a template of a variant, see section *Product Variant and Master Product*.

2.3.1 Configure the Specification Section

The *Specification* section can be configured to generate dynamic content and simplify product information management.

For more information about configuring the *Specification* section, see *the Sitecore E-Commerce API Reference Guide*.

2.4 Creating Products and Product Categories in a Product Repository

You can store the product information for all the products you want to sell on your webshop in the *Product Repositories* folder.

By default, SES uses Sitecore CMS content to store product information. However, you can customize your installation to get the product information from other software products.

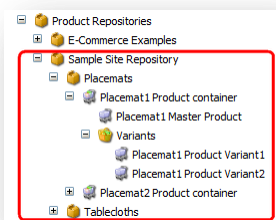
Even though you can store the product information outside of Sitecore CMS, we recommend that you store it in Sitecore CMS. If you store the product information in Sitecore, you can use various integrated Sitecore modules and extensions. For example, you can use Sitecore DMS for a number of marketing purposes, such as personalization, marketing goals and so on. For more information about integrating SES with Sitecore DMS, read the *SES DMS Cookbook*.

Different search providers can be used to index the products in your webshops. For more information about configuring search providers, see the section *Configuring the Website Backend Search*.

2.4.1 The Optimal Webshop Repository Structure

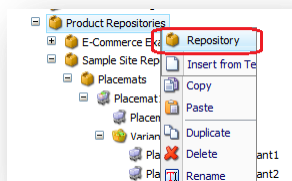
When you create products in SES, you must create an item for each product, because the search provider assumes that the product information is stored in the corresponding Sitecore item.

Even though E-Commerce allows greater flexibility when structuring your webshop products in categories, we recommend that you use the following structure as a model when you build your custom webshop.

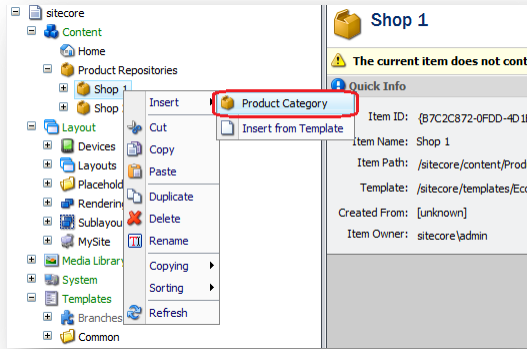


The ideal structure of a webshop must ensure that:

- All the webshops are placed under the *Product Repositories* item. In our example, there are two webshops under the *Product Repositories* item: *E-Commerce Examples* and *Sample Site Repository*.
- Each webshop is based on the *Repository* template.



- Each product category is based on the integrated *Product Category* template. For more information about how to create a product category using the *Product Category* template, see the section *Creating Product Categories in a Product Repository*.

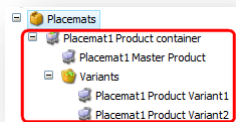


- You can create as many nested categories as you wish.
- A product item contains all the information about this particular product.

Important

In SES, we recommend that you store all the information about a product in the product item. SES cannot incorporate the information about the product, if this information is placed in its category folders or in any external item.

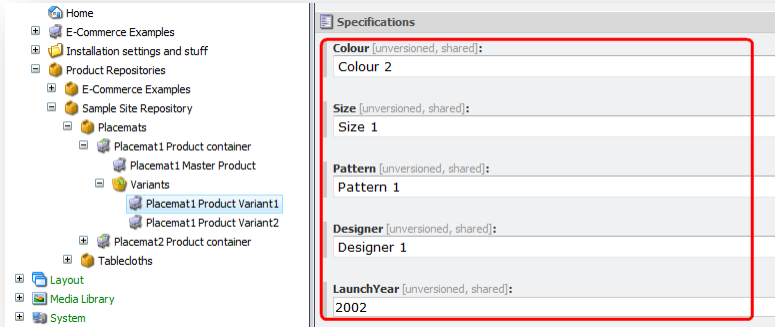
- Each product is represented as a master product and its variants. For more information about master products and variants, read the section *Product Variant and Master Product*.
- Products should be stored under the product category to copy the following structure:
 - Product folder
 - Master product
 - Variants folder
 - Product variant 1
 - Product variant 2
 - <...>



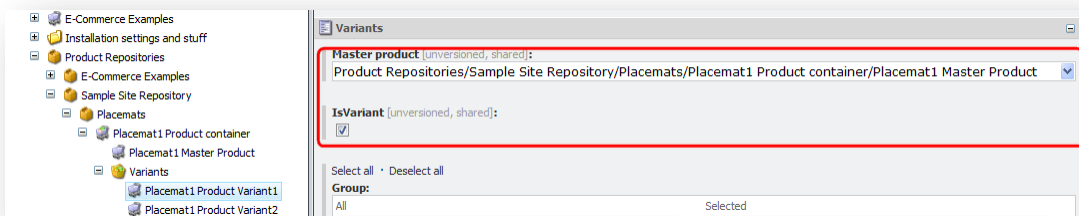
2.4.2 Product Variant and Master Product

Your webshop may need to sell different variations of the same product, for example, a shoe can come in different colors and sizes. To support this scenario, you can implement product variants.

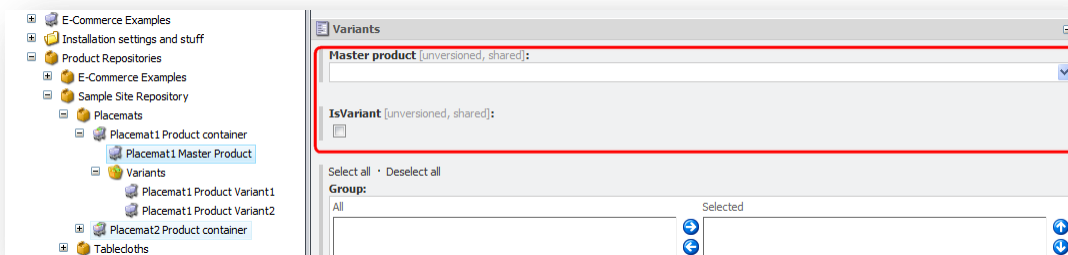
A product variant is based on the same template on which the variant's master product is based. A product variant is a copy of the product in which one or several fields have different values than those in the product.



Product variants could have an extra field that contains a link to the master product.



A master product is the unique product variant that represents the product on the web pages. The master product itself has no links to variants.



All the product variants of a product type refer to the same master product.

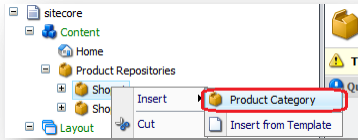
For more information about product variants and how to implement them, see *the SES Configuration Guide*.

2.4.3 Creating Product Categories in a Product Repository

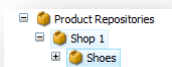
Normally, products are grouped into product categories. For example, the products boots and slippers belong to one category — shoes. The *Product Repositories* item allows you to create a nested structure of product categories.

To create a product category:

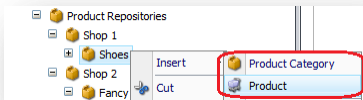
1. Create a product category item under your webshop repository.



2. Give the category a suitable name, for example, *Shoes*.

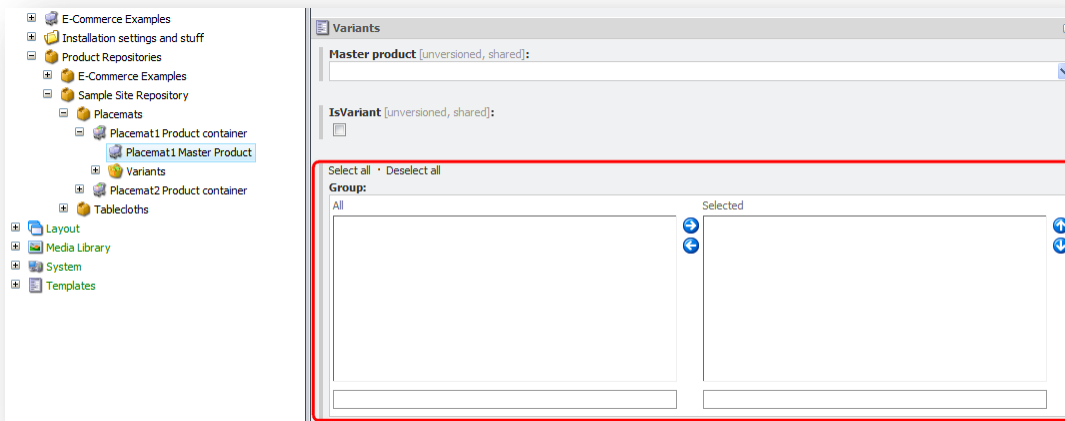


3. If you right-click the *Shoes* item and click **Insert**, you can either create another product category inside the existing product category, for example, *Boots*, or create a product item.



Note

SES cannot retrieve product information from products that are structured in categories. Structuring products in category folders just improves your overview of the products but it is not a recommended approach. If you want to find a product that belongs to a category, you must create a product category field on the product template. For more information about retrieving product information, see the *SES Developer's Cookbook*.

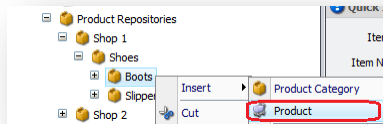


2.4.4 Creating Products Items in a Product Repository

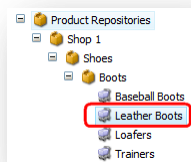
After you create the product categories, you can also create the individual product items. In this example, you create a product item called *Leather Boots* that is based on the *Product* template.

To create a product item:

1. In the content tree, right-click the product category in which you want to create this product item, click *Insert*, and then click *Product*.



2. Give the category a suitable name, for example, *Leather Boots*.
3. The new product appears in the product category.



You can create as many products in a product category as you want.

Extending the Product Template

Depending on the variety of your products, you can create products for your webshop based on one template or on several product templates.

You can modify these templates by adding field sections and fields.

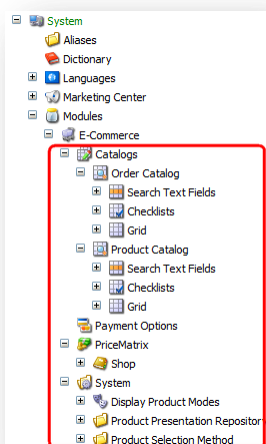
For more information about how to create or extend product templates, see the section *Creating Custom Product Templates*.

Configuring Product Prices

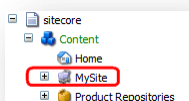
You can configure product prices in SES. However, in the current version, the price matrix is hard coded. If you want to extend the price model, you must change the code programmatically.

2.5 Creating the Root Folder for your Webshop

You must create the root folder to store the website of your webshop and *most of the* webshop settings. The remaining webshop settings are stored in the `Sitecore/System/Modules/E-Commerce` folder.



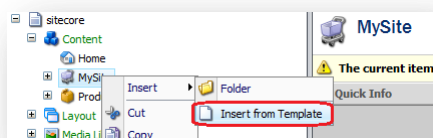
To create the root folder, create a folder below the `Sitecore/Content/` item and give it a suitable name.



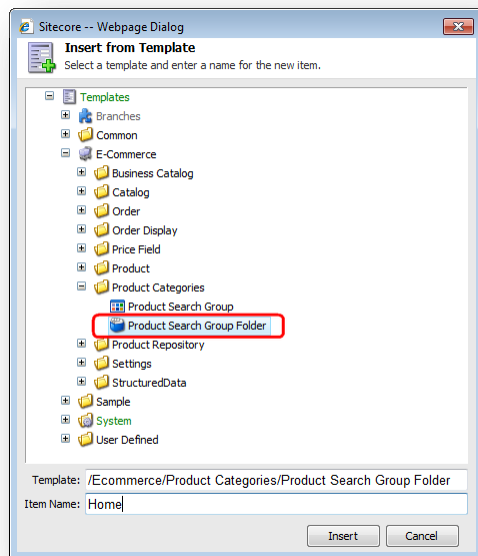
Now you need to create a website root node to make your webshop resolve products.

To create the website root node for your webshop:

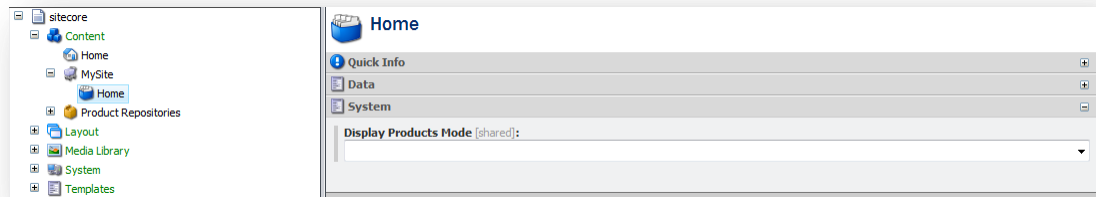
1. Right-click the `Sitecore/Content/MySite` item, click **Insert** and then click **Insert from Template**.



2. In the **Insert from Template** dialog box, navigate to the **Product Search Group Folder** template.



3. In the **Insert from Template** dialog box, in the **Item Name** field, enter *Home* and click **Insert**.



Note

By creating a website root node, you create a fallback for resolving products. The important field on that template is the **System** field. The **System** field defines the default way to resolve products. By being able to configure the **Display Products Mode** setting, you can make all the products of your webshop displayed in one way.

Chapter 3

Configuring Settings

This chapter describes how to configure the Webshop Site Settings, a Webshop Site Settings item, the Business Catalog Settings and the General Settings of your webshop.

The chapter contains the following sections:

- Webshop Site Settings
- Business Catalog Settings
- General Settings

3.1 Webshop Site Settings

You must use webshop site settings to specify how the various elements in your website appear on the page and to determine some of the information that they display. Some of the settings determine how the different elements of your webshop interact with one another.

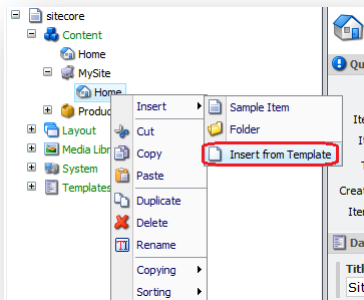
3.1.1 Creating Webshop Site Settings Item

By default, SES looks for the *Webshop Site Settings* directly under the root of the website definition. If you store the Webshop Site Settings somewhere else, or if they are named differently, you must define a special attribute for Sitecore E-Commerce. The attribute for the website definition is called `EcommerceSiteSettings="/Site Settings"`. This attribute indicates the relative path to the Webshop Site Settings folder in relation to the site root (start item of the site). For more information about the *Webshop Site Settings* attribute, see the section *Registering Your Website*.

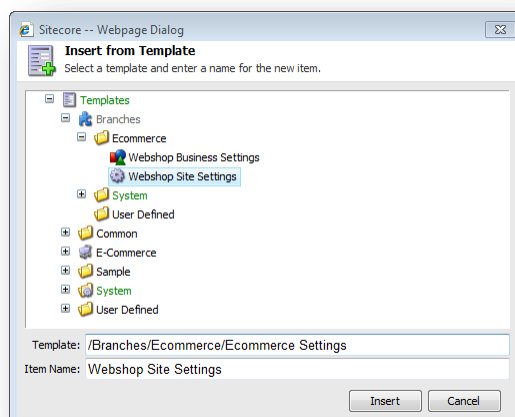
To include the site settings in one folder, you must create a *Webshop Site Settings* item.

To create a *Webshop Site Settings* item:

1. Right-click the `Sitecore/Content/MySite/Home` item, click **Insert**, and then click **Insert from Template**.

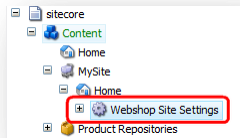


2. In the **Insert from Template** dialog box, navigate to the **Webshop Site Settings** template.

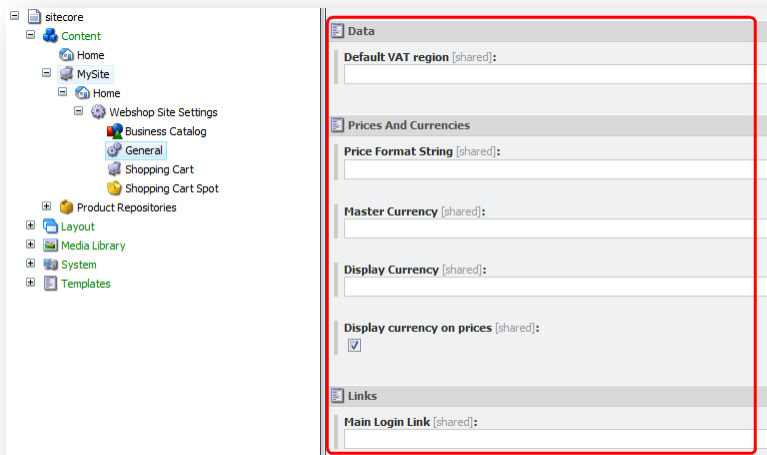


3. In the **Insert from Template** dialog box, in the **Item Name** field, enter *Webshop Site Settings* and click **Insert**.

4. A new *Webshop Site Settings* item appears under the *Home* item.



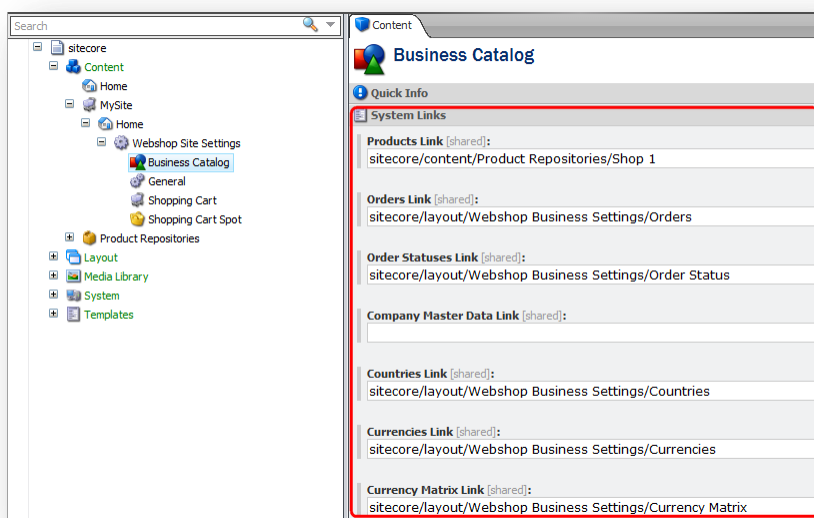
You can edit these site settings items.



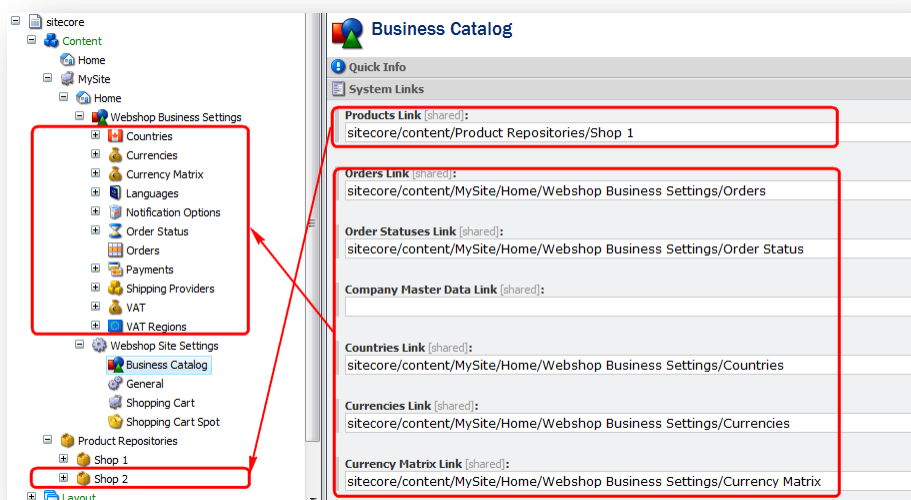
For more information about the webshop site settings, see the *SES Configuration Guide*.

3.2 Business Catalog Settings

You must use the *Business Catalog* item settings to define the path to the different items in the Business Catalog.



Each field in the *Business Catalog* item contains a link to the corresponding item in the Webshop Business Settings.

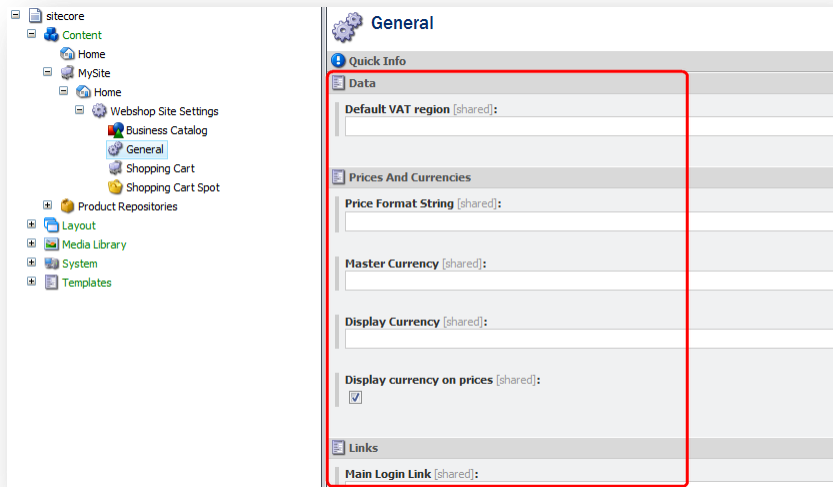


The Webshop Business Settings items can be stored in different locations. You must use the drop down lists in the fields to select the alternative locations of the Business Catalog items.

The **Product Link** field contains the path to the folder where your products are stored and has nothing to do with the Webshop Business Settings.

3.3 General Settings

You must use the *General* item to specify more settings for your webshop.



These settings include the default VAT region that your webshop is in, the currency used on the webshop, and the role assigned to customers on your webshop.

The *General* item contains the following fields:

Field Name	Field Type	Description
Default VAT Region	Droptree	It represents the VAT region that the webshop is in.
Price Format String	Single-Line Text	It represents the format in which prices are displayed (except for the Shopping Cart page and the Shopping Cart spot. For more information about how to edit the Price Format String field, see the following web links: http://msdn.microsoft.com/en-us/library/txafckwd(v=VS.90).aspx http://www.csharp-examples.net/string-format-double/
Master Currency	Droplink	It represents the currency in which a product is priced in the shop.
Display Currency	Droplink	It represents the currency in which the price of a product is displayed on the webshop pages.
Display Currency on Prices	Check box	Select this check box, to display the currency along with the price.
Main Login Link	Droptree	It represents the link to the <i>Login</i> page. You must use this setting to define customer account pages.
Password Reminder Link	Droptree	It represents the link to the <i>Reset Password</i> page. You must use this setting to define customer account pages.
New User Account Link	Droptree	It represents the link to the <i>Create New Account</i> page. You must use this setting to define

Field Name	Field Type	Description
		customer account pages.
My Page Link	Droptree	It represents the link to the page called <i>My Page</i> . You must use this setting to define customer account pages.
Search Page Link	Droptree	It represents the link to the <i>Search Results</i> page. You must use this setting to define customer account pages.
Mail Templates Link	Droptree	It represents the link to the mail templates folder that contains the templates for all the e-mails used on the webshop.
Default Customer Roles	Single-Line Text	In this field, enter the names of the security roles that you want to assign to visitors to your webshop who have created a customer account. If there is more than one role, enter them in a pipe (' ') separated list.

Note

If you change the *Price Format String* setting for the *General* item, it will affect the way the prices are displayed on all the pages in your webshop except for the *Shopping Cart* page and the *Shopping Cart* spot.

Chapter 4

Website

This chapter describes how to register your website and build your webshop pages.

The chapter contains the following sections:

- Registering Your Website
- Building the Webshop Pages
- Configuring the Frontend Search on the Webshop

4.1 Registering Your Website

You must register your website.

You register a website so that it is displayed when you type the URL without `/sitecore` or just type a specific hostname. When you register a website, you specify where SES can find the website's Site Settings.

You can register your website by either adding an additional inclusion config file to the corresponding folder of your webshop (preferred way) or edit the `web.config` file directly.

If you want to register your website by editing the `web.config` file:

1. Open the `web.config` file of the website where you have installed the E-Commerce webshop core package.
2. Navigate to the `<site name="website" setting`.

```
<sites>
  <site name="shell" virtualFolder="/sitecore/shell" physicalFolder="/sitecore/shell" rootPath="/sitecore/content" startItem="/home" language="en" database="core" enableAnalytics="false" database="core" domain="sitecore"/>
  <site name="login" virtualFolder="/sitecore/login" physicalFolder="/sitecore/login" enableAnalytics="false" database="core" domain="sitecore"/>
  <site name="testing" virtualFolder="/sitecore/testing" physicalFolder="/sitecore/testing" rootPath="/sitecore/content" enableAnalytics="false" enableWorkflow="true" domain="sitecore" loginFolder="ce"/>
  <site name="admin" virtualFolder="/sitecore/admin" physicalFolder="/sitecore/admin" enableAnalytics="false" enableWorkflow="true" domain="sitecore" loginFolder="ce"/>
  <site name="service" virtualFolder="/sitecore/service" physicalFolder="/sitecore/service"/>
  <site name="modules_shell" virtualFolder="/sitecore modules/shell" physicalFolder="/sitecore modules/shell" rootPath="/sitecore/content" startItem="/home" language="en" database="core" domain="sitecore"/>
  <site name="modules_website" virtualFolder="/sitecore modules/web" physicalFolder="/sitecore modules/web" rootPath="/sitecore/content" startItem="/home" language="en" database="core" domain="sitecore"/>
  <site name="website" virtualFolder="/" physicalFolder="/" rootPath="/sitecore/content" startItem="/EcommerceSiteSettings" language="en" database="core" domain="sitecore"/>
  <site name="scheduler" enableAnalytics="false" domain="sitecore"/>
  <site name="system" enableAnalytics="false" domain="sitecore"/>
  <site name="publisher" domain="sitecore" enableAnalytics="false" enableWorkflow="true"/>
</sites>
```

3. In the same line, change the `startItem` parameter to point to the root item of your webshop, in this case `MySite`.

```
<sites>
  <site name="shell" virtualFolder="/sitecore/shell" physicalFolder="/sitecore/shell" rootPath="/sitecore/content" startItem="/home" language="en" database="core" enableAnalytics="false" database="core" domain="sitecore"/>
  <site name="login" virtualFolder="/sitecore/login" physicalFolder="/sitecore/login" enableAnalytics="false" database="core" domain="sitecore"/>
  <site name="testing" virtualFolder="/sitecore/testing" physicalFolder="/sitecore/testing" rootPath="/sitecore/content" enableAnalytics="false" enableWorkflow="true" domain="sitecore" loginFolder="ce"/>
  <site name="admin" virtualFolder="/sitecore/admin" physicalFolder="/sitecore/admin" enableAnalytics="false" enableWorkflow="true" domain="sitecore" loginFolder="ce"/>
  <site name="service" virtualFolder="/sitecore/service" physicalFolder="/sitecore/service"/>
  <site name="modules_shell" virtualFolder="/sitecore modules/shell" physicalFolder="/sitecore modules/shell" rootPath="/sitecore/content" startItem="/home" language="en" database="core" domain="sitecore"/>
  <site name="modules_website" virtualFolder="/sitecore modules/web" physicalFolder="/sitecore modules/web" rootPath="/sitecore/content" startItem="/home" language="en" database="core" domain="sitecore"/>
  <site name="website" virtualFolder="/" physicalFolder="/" rootPath="/sitecore/content" startItem="/MySite" EcommerceSiteSettings="/Webshop Site Settings" language="en" database="core" domain="sitecore"/>
  <site name="scheduler" enableAnalytics="false" domain="sitecore"/>
  <site name="system" enableAnalytics="false" domain="sitecore"/>
  <site name="publisher" domain="sitecore" enableAnalytics="false" enableWorkflow="true"/>
</sites>
```

4. In the same line, insert the default setting `EcommerceSiteSettings`. The setting value must contain the path to your *Webshop Site Settings* folder.

```
otPath="/sitecore/content" startItem="/home" language="en" database="core" enableAnalytics="false" database="core" domain="sitecore"/>
  <site name="login" virtualFolder="/sitecore/login" physicalFolder="/sitecore/login" enableAnalytics="false" database="core" domain="sitecore"/>
  <site name="testing" virtualFolder="/sitecore/testing" physicalFolder="/sitecore/testing" rootPath="/sitecore/content" enableAnalytics="false" database="core" domain="sitecore" loginFolder="ce"/>
  <site name="admin" virtualFolder="/sitecore/admin" physicalFolder="/sitecore/admin" enableAnalytics="false" enableWorkflow="true" domain="sitecore" loginFolder="ce"/>
  <site name="service" virtualFolder="/sitecore/service" physicalFolder="/sitecore/service"/>
  <site name="modules_shell" virtualFolder="/sitecore modules/shell" physicalFolder="/sitecore modules/shell" rootPath="/sitecore/content" startItem="/home" language="en" database="core" domain="sitecore"/>
  <site name="modules_website" virtualFolder="/sitecore modules/web" physicalFolder="/sitecore modules/web" rootPath="/sitecore/content" startItem="/home" language="en" database="core" domain="sitecore"/>
  <site name="website" virtualFolder="/" physicalFolder="/" rootPath="/sitecore/content" startItem="/MySite" EcommerceSiteSettings="/Webshop Site Settings" language="en" database="core" domain="sitecore"/>
  <site name="scheduler" enableAnalytics="false" domain="sitecore"/>
  <site name="system" enableAnalytics="false" domain="sitecore"/>
  <site name="publisher" domain="sitecore" enableAnalytics="false" enableWorkflow="true"/>
</sites>
```

Note

You must configure the Site Settings, because they point to everything that is related to that particular website.

For more information about how to register a website, see the article [Configuring Sites in the web.config File](#) on the SDN.

4.1.1 Order Manager Site Context Name

For information on setting CMS Core and Order Manager web site definitions and site context name, see the *Post Installation Steps* section of the *SES Installation Guide* on SDN.

4.2 Building the Webshop Pages

To build your webshop pages:

- Create the product category pages and the product category folders.
- Add the products from the Product Repositories to the webshop pages.
- Configure the Product Catalog search options of the pages that you want to display the products on.
- Configure the way the products are presented on the page.

4.2.1 Creating the Product Category Pages

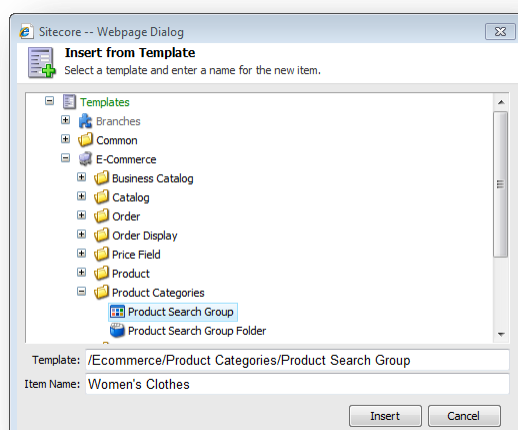
A webshop normally presents the products to the customers in categories. For example, if you sell clothes, your webshop will have categories for different types of clothes.

In SES, there is a distinction between the way products are organized in the product repository and the way products are presented on the website. You can use product category pages to build website pages that display products in categories to website visitors.

Creating a Product Category Page

To create a product category page:

1. In the content tree, right-click the `Sitecore/Content/MySite/Home` item, click **Insert** and then click **Insert from Template**.

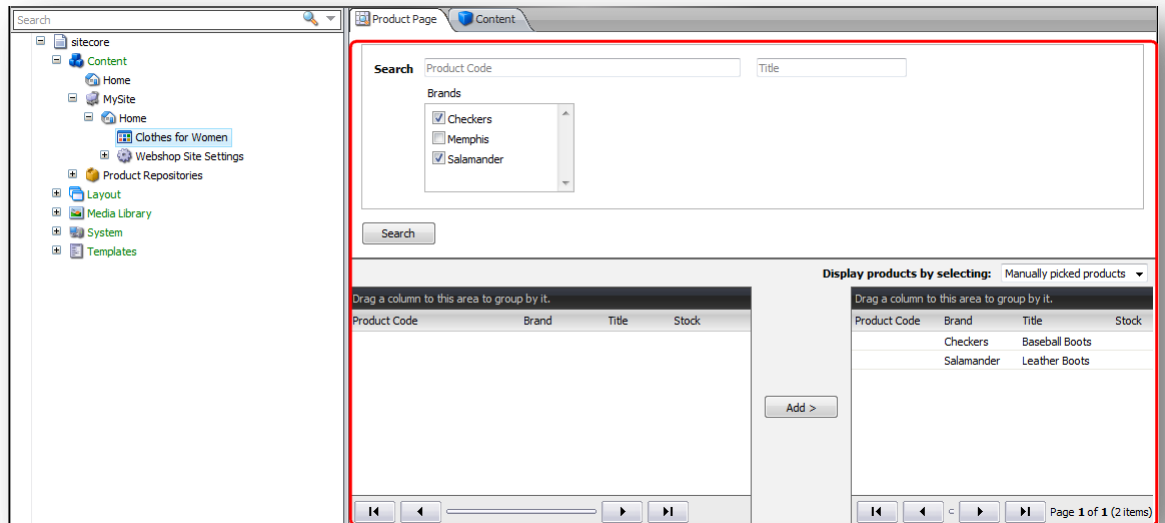


2. In the **Insert from Template** dialog box, navigate to the `Sitecore/Templates/E-Commerce Product Categories` folder, and select the *Product Search Group* template.
3. In the **Item Name** field, enter the name of the category page, click **Insert** and the product category page appears under the *Home* item.

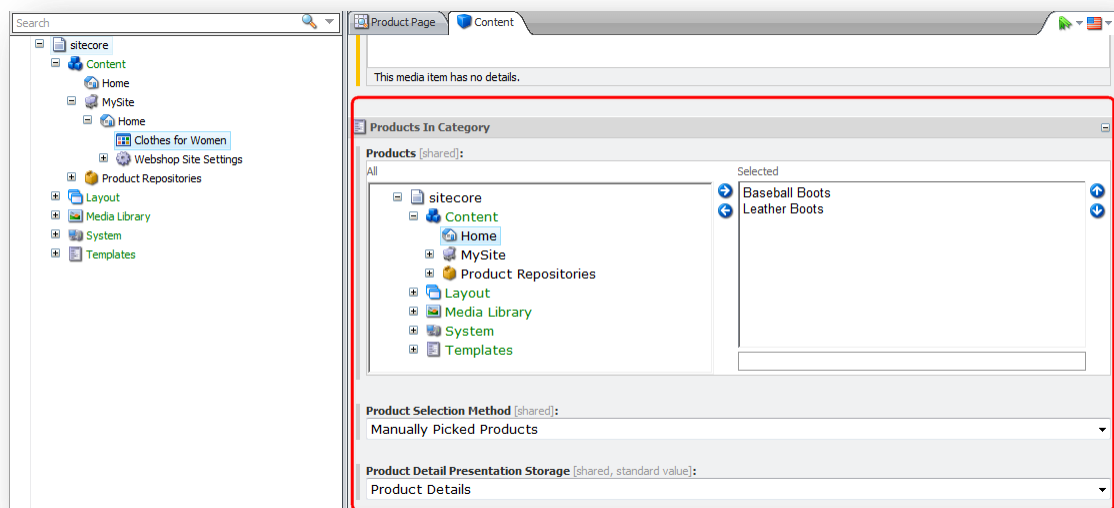
Note

The *Product Category* template can be used as is, without any modifications. But it can also be inherited and extended.

4. On the **Product Page** tab of the new product category page, you can specify whether you want the page to display the results of a search or to display the products that you pick manually.



5. On the **Content** tab, you can define some other properties of the new webshop page.



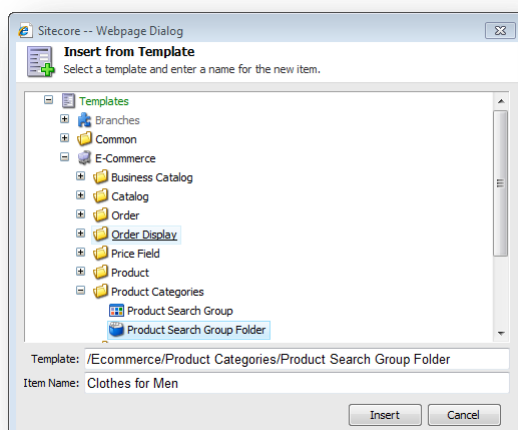
Creating a Product Category Folder

A product category folder is a folder item that can contain several product category pages. On the product category folder, you can define additional settings that apply to all the subordinate product category pages.

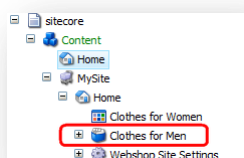
To create a product category folder:

1. Right-click the Sitecore/Content/MySite item, click **Insert**, and then click the **Insert from Template** option.

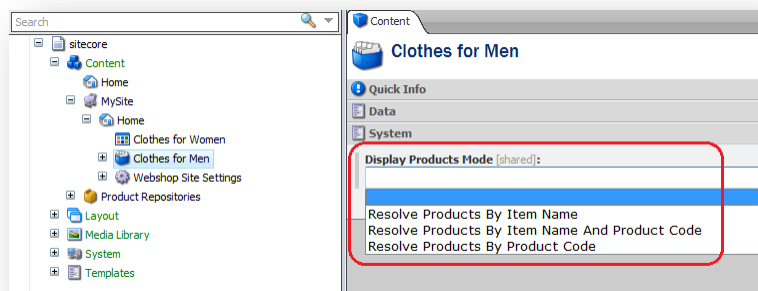
- In the **Insert from Template** dialog box, navigate to the *Sitecore/Templates/E-Commerce Product Categories* folder and select the *Product Search Group Folder* template.



- In the **Item Name** field, enter the name of the category page and click **Insert**. The product category page appears under the *Home* item.



- In the product category folder item, you define the *Display Products Mode* system setting that is not available on the product category page.



For more information about how to edit the fields on the **Content** tab, see the *SES Configuration Guide*.

4.2.2 Adding Repository Products to Webshop Pages

Now that you have created the product category page, you can add products to it.

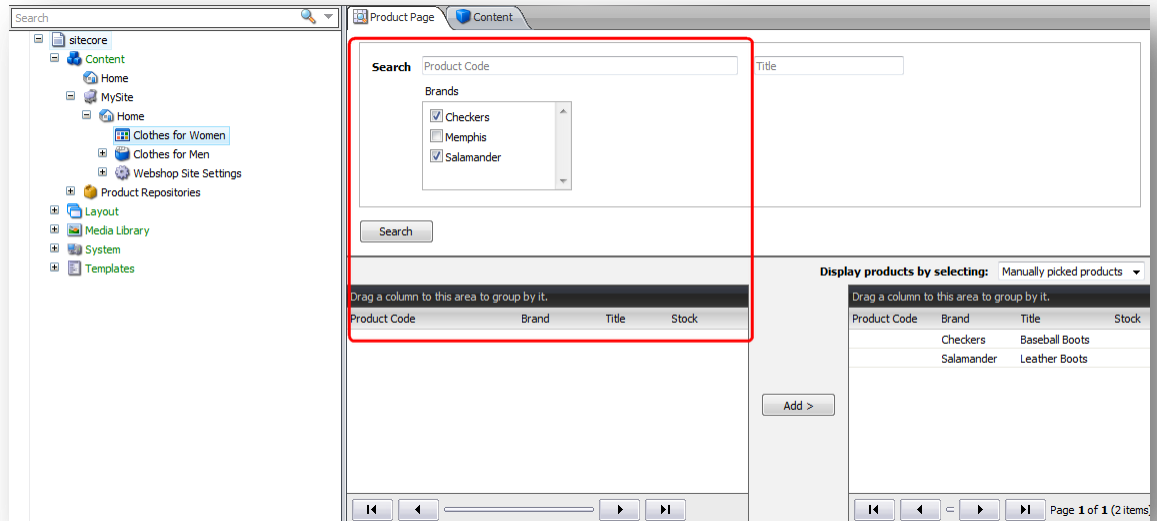
Note

When you add a product to the product category page, you do not add actual products; you only add references to the actual products which are stored under the *Product Repositories* item. If you use the Lucene search provider to search for products, it will also find these product references and not only the actual products from the Product Repositories.

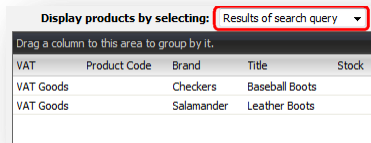
You can only edit products on the product category pages that contain the Product Catalog. You can use the search results to edit the product selection automatically on the **Product Page** tab or you can select the products manually.

To add products using the search results:

1. On the **Product Page** tab of the page that you want to add products to, search for the products you want.



2. In the **Display products by selecting** dropdown list, select the *Results of search query* option.



3. Click **Save** to save your selection.

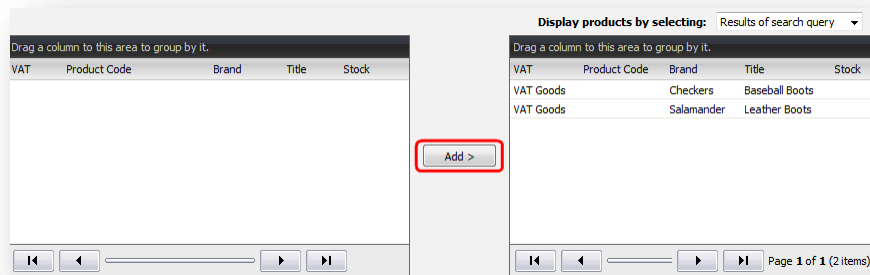
Note

You add all of the products that correspond to the search criteria when you add products using the *Results of search query* option. In other words, if more products with the specified search options appear in the Product Repositories, these products are automatically added to the product category page.

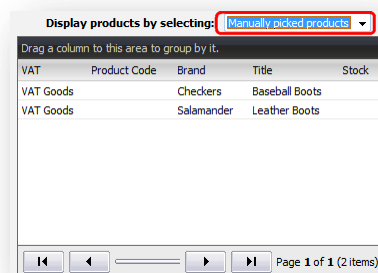
To add products manually:

1. On the page where you want to add products, on the **Product Page** tab, search for the products that you want to display on your website.

- Click **Add** to move the products from the left column to the right column.



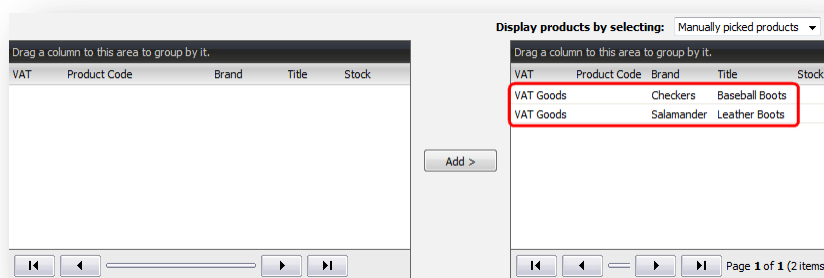
- In the **Display products by selecting** dropdown list, select the *Manually picked products* option.



- Click **Save**.

Note

When you use the *Manually picked products* option, you only add the products that you selected to the product category page. Only the products in the right-hand column are displayed on the product category page.



For more information about how to extend the search criteria, see the *Sitecore E-Commerce API Reference Guide*.

4.3 Configuring the Frontend Search on the Webshop

The list of products displayed on the site and that in the repository are not necessarily the same. This search is to help customers find a product that is displayed on the site.

The following steps explain how to set up the search in SES.

1. Configure the Lucene index.

As indicated in the following code snippet, create a new config file and call it `Ecommerce.Sample.config`

```
<?xml version="1.0"?>
<configuration xmlns:patch="http://www.sitecore.net/xmlconfig/">
  <sitecore>
    <search>
      <configuration>
        <indexes>
          <index id="web" type="Sitecore.Search.Index, Sitecore.Kernel">
            <param desc="name">web</param>
            <param desc="folder">__web</param>
            <Analyzer type="Sitecore.Ecommerce.Search.LuceneAnalyzer,
Sitecore.Ecommerce.Kernel" />
            <locations hint="list:AddCrawler">
              <web type="Sitecore.Ecommerce.Search.VirtualProductsCrawler,
Sitecore.Ecommerce.Kernel">
                <Database>web</Database>
                <Root>{110D559F-DEA5-42EA-9C1C-8A5DF7E70EF9}</Root>
                <Tags>web</Tags>
              </web>
            </locations>
          </index>
        </indexes>
      </configuration>
    </search>
  </sitecore>
</configuration>
```

Give suitable names to the search index and the folder, Insert the GUID of the home page of your website.

Note

If you are using Lucene, you must configure the search index. This step is not mandatory for the other search providers.

2. Create a custom frontend search class.

You can use the following code snippet to complete this step:

```
using System.Collections.Generic;
using Sitecore;
using Sitecore.Diagnostics;
using Sitecore.Ecommerce;
using Sitecore.Ecommerce.Catalogs;
using Sitecore.Ecommerce.Search;

namespace SearchSample
{
  public class FrontEndSearch
  {
    /// <summary>
    /// The search provider
    /// </summary>
    private ISearchProvider searchProvider;

    /// <summary>
    /// Gets or sets the product resolver.
    /// </summary>
    /// <value>The product resolver</value>
```



```

protected VirtualProductResolver ProductResolver { get; set; }

/// <summary>
/// Initializes a new instance of the <see cref="ProductRepository"/> class.
/// </summary>
public FrontEndSearch() : this(new SitecoreQuerySearchProvider()){}

/// <summary>
/// Initializes a new instance of the <see cref="ProductRepository"/> class.
/// </summary>
/// <param name="searchProvider">The search provider.</param>
public FrontEndSearch([NotNull] ISearchProvider searchProvider)
{
    Assert.ArgumentNotNull(searchProvider, "searchProvider");

    // Setting current searchprovider to use.
    this.searchProvider = searchProvider;

    // Resolving the VirtualProducts Resolver
    this.ProductResolver =
Sitecore.Ecommerce.Context.Entity.Resolve<VirtualProductResolver>();
}

/// <summary>
/// Method for applying the actual search
/// </summary>
/// <param name="searchWords">The search words to search for</param>
/// <returns>List of results as items</returns>
public IEnumerable<Sitecore.Data.Items.Item> Search(string searchWords)
{
    // Creating a new Query for searching the Lucene catalog
    Sitecore.Ecommerce.Search.Query query = new
Sitecore.Ecommerce.Search.Query();

    // Setting the Search Root to the Start Item of the current site - this
    // must be set in site settings in the Ecommerce.Sample.config file
    query.SearchRoot =
Sitecore.Context.Database.GetItem(Sitecore.Context.Site.StartPath).ID.ToString();

    // Adding fields to the query in the catalog
    query.Add(new FieldQuery("Title", searchWords, MatchVariant.Exactly));

    // Searching
    IEnumerable<Sitecore.Data.Items.Item> searchResult =
searchProvider.Search(query);

    // Returning search result
    return searchResult;
}
}
}

```

3. Register your custom frontend search class in Unity and configure it.

To ensure that the search is generic and open to any search engine, you must add the following code snippet to the `Unity.config` file under the `<Unity>` node:

```
<alias alias="FrontEndSearch" type="SearchSample.FrontEndSearch, SearchSample" />
```

You must also add the following snippet under the `<container>` node:

```

<register type="FrontEndSearch" mapTo="FrontEndSearch" >
  <lifetime type="perthread" />
  <constructor>
    <param name="searchProvider">
      <!--<dependency name="FastQuerySearchProvider" />-->
      <dependency name="LuceneFrontEndSearchProvider" />
    </param>
  </constructor>
</register>

```

```
<register type="ISearchProvider" mapTo="LuceneSearchProvider"
name="LuceneFrontEndSearchProvider">
  <property name="IndexName" value="web" />
</register>
```

As shown in the last snippet you can configure the search to use another provider such as *FastQuerySearchProvider*.

4. Create a sub-layout to call the frontend search class.

You need to create at least a text box that the webshop visitor can enter search keywords into. Give the test box a suitable name, for example `bodySearch`. You must also create a button to execute the search.

You also need to create a code behind file to read the search keywords that the visitors enter and call the frontend search methods.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using SearchSample;
using Sitecore.Ecommerce;

namespace Sitecore.Ecommerce.WebSite.layouts
{
    public partial class MySearchResult : System.Web.UI.UserControl
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!this.Page.IsPostBack)
            {
                if ((HttpContext.Current.Request["search"] ?? string.Empty) !=
string.Empty)
                {
                    string searchWords =
UrlDecode((HttpContext.Current.Request["search"] ?? string.Empty));
                    FrontEndSearch frontEndSearch =
Sitecore.Ecommerce.Context.Entity.Resolve<FrontEndSearch>();

                    IEnumerable<Sitecore.Data.Items.Item> items =
frontEndSearch.Search(searchWords);
                    foreach (Sitecore.Data.Items.Item item in items)
                    {
                        // Simple output
                        Response.Write(item.Name + "<br>");
                    }
                }
                else if (SearchAgainClicked() && (this.bodySearch.Value != string.Empty))
                {
                    base.Response.Redirect("/MySearch.aspx?search=" +
this.bodySearch.Value);
                }
                private static bool SearchAgainClicked()
                {
                    return (HttpContext.Current.Request["SearchAgain"] ??
string.Empty).Equals("SearchAgain");
                }

                public static string UrlDecode(string url)
                {
                    if (url == null)
                    {
                        return string.Empty;
                    }
                    return HttpUtility.UrlDecode(url);
                }
            }
        }
    }
}
```

Chapter 5

Product Presentation

This chapter describes how to configure the Order Catalog and the Product Catalog of your webshop. You will learn how to create search options lists in both of these catalogs. Moreover, you will be able to configure various presentation settings, such as display product modes, and product selection method and the product details item.

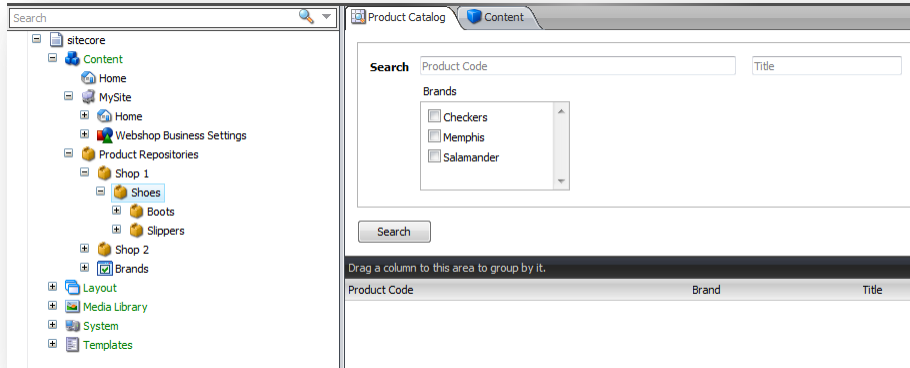
The chapter contains the following sections:

- Configuring the Product Catalog
- Configuring the Presentation Settings

5.1 Configuring the Product Catalog

The Product Catalog belongs to the built-in functionality of the webshop.

You must use the Product Catalog to search for products.

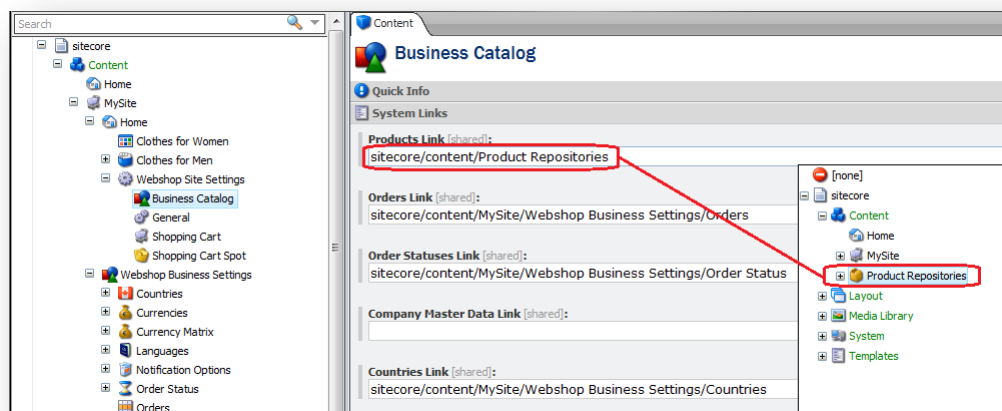


You can customize the search options for the product catalog.

Product Catalog Configuration Prerequisites

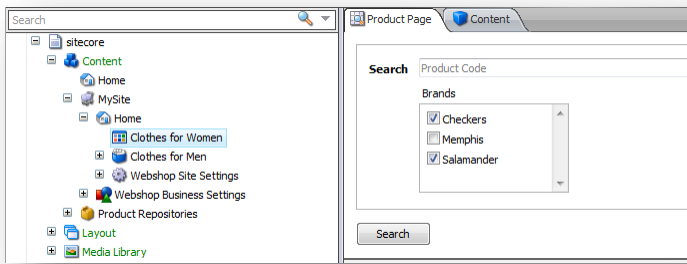
Before you start to configure the *Product Catalog*:

1. Make sure that the website is registered.
For more information about how to register the E-Commerce website, see the section *Registering Your Website*.
2. Make sure that you have implemented the search settings.
For more information about how to configure search, see the section *Configuring the Website Backend Search*.
3. Make sure that the **Products Link** field in the Webshop Site Settings/Business Catalog points to the *Product Repositories* item.



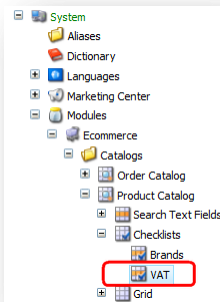
Creating a Search Options List

Apart from the default *Brands* search options list in the *Product Catalog*, you can create other search options lists or check box lists.

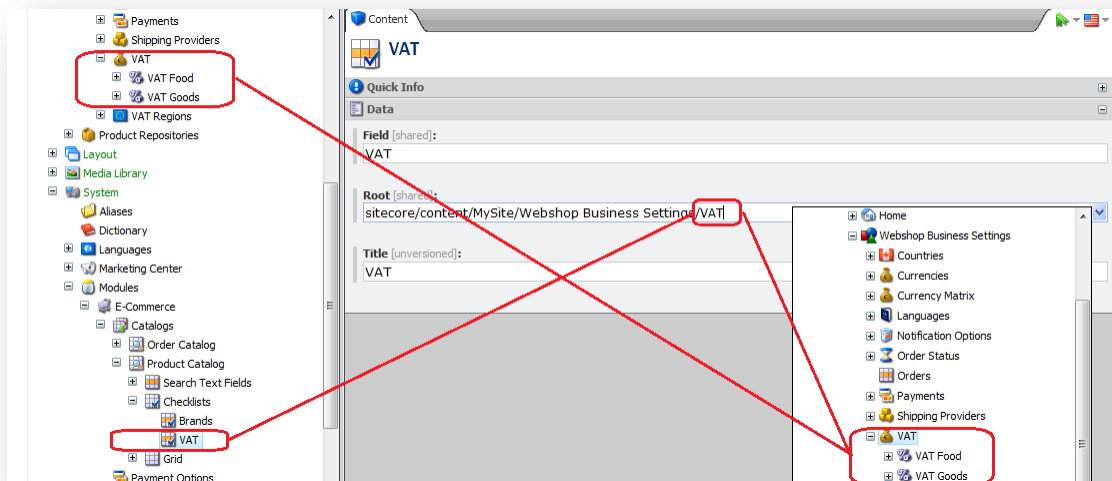


To create a search options list:

1. In the **Page Editor**, in the *System/Modules/E-Commerce/Catalogs/Product Catalog/Checklists* folder, right-click the *Brands* item and duplicate the *Brands* item to create a *VAT* checklist item.

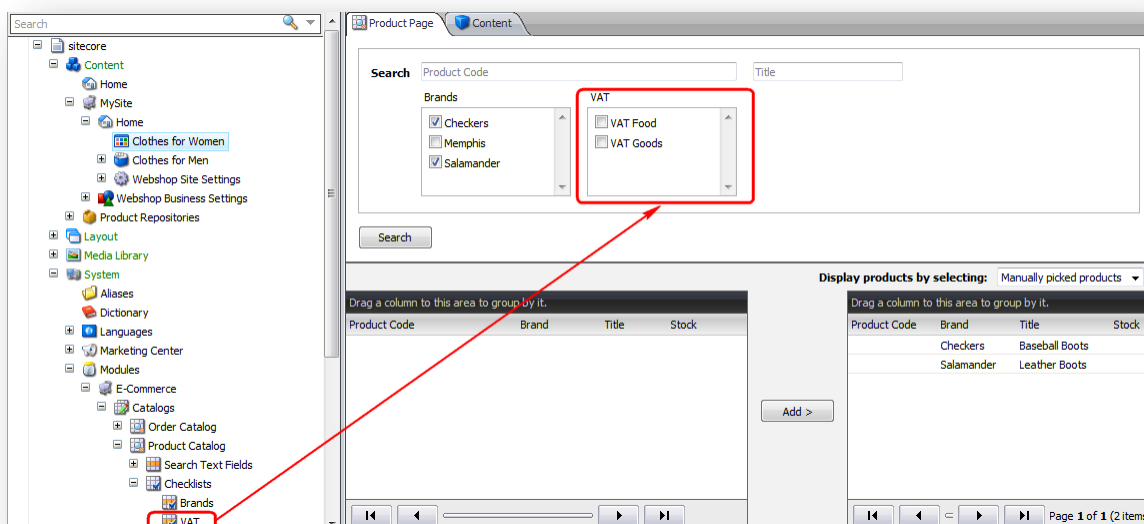


2. In the field called **Field**, enter the name of the item and save the item.
3. In the **Root** field, click the drop-down arrow and select the *sitecore/content/MySite/Webshop Business Settings/VAT* item.

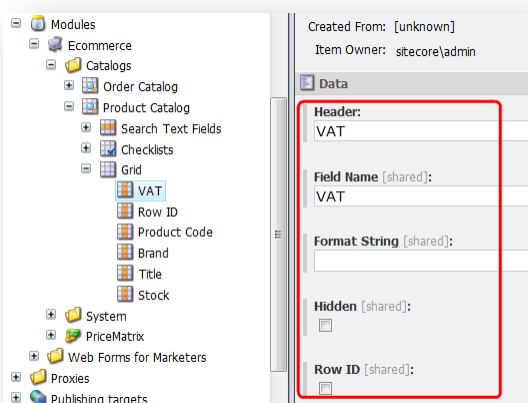


The `sitecore/content/MySite/Webshop Business Settings/VAT` item points to the *VAT Food* and *VAT Goods* sub items. This means that you can search either for the *VAT Food* or for the *VAT Goods* option in the *VAT* search options list.

4. Save the changes.
5. Click a product category page and you can see that a new check box list has been added to the search page.



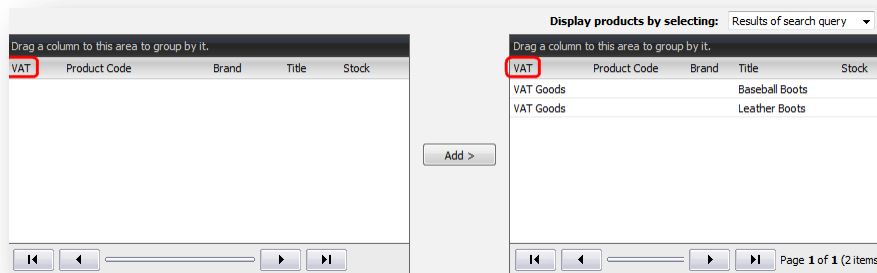
6. After you have added a check box list to the *Product Catalog*, you need to ensure that the name of the field appears in the search results list. The *VAT* column in the search results list displays the *VAT* option that you are searching for.
7. Create a *VAT* item in the `sitecore/system/Modules/Ecommerce/Catalogs/Product Catalog/Grid` folder.
8. Ensure that the fields in the *VAT* sub item contain the following information:



For more information about how to fill in the *VAT* item fields, see the *SES Configuration Guide*.

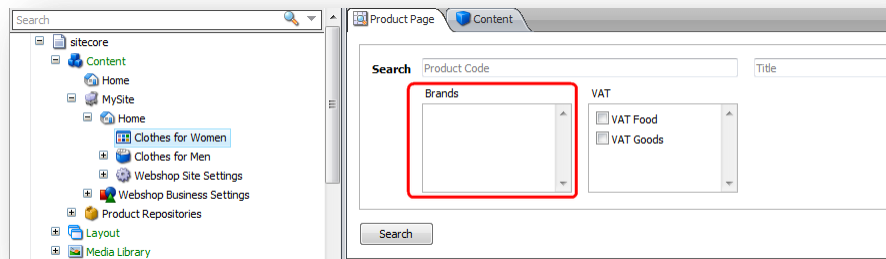
9. Save the changes.

Go back to the product category, the search results list on the *Product Catalog* page display the *VAT* column.



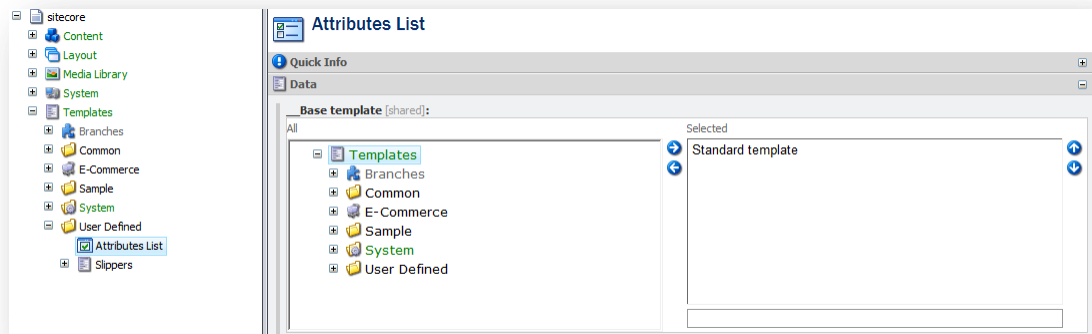
Configuring Search Options in an Existing Checkbox List

You may not want to create a new checkbox list, but fill out the existing checkbox list, for example the *Brands* one, with search options.

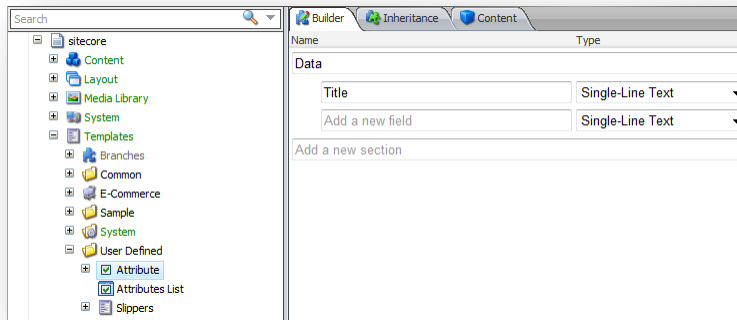


To create search options:

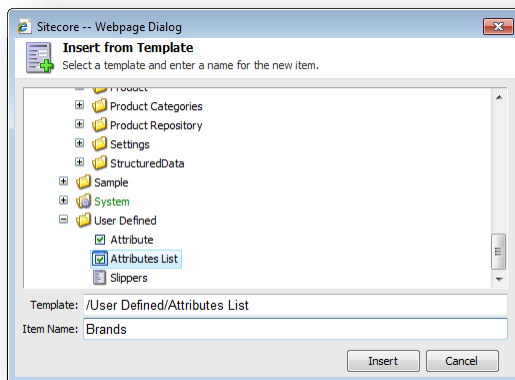
1. In the *Templates/User Defined* folder, create an *Attribute List* template that inherits the **Standard template**.



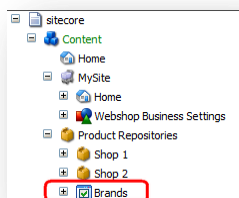
- Ensure that this template also includes the *Data* field section and the *Title* field.



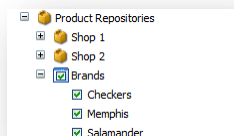
- Under the *Product Repositories* item, create a *Brands* item based on the *Attributes List* template. To do this, right-click the `Sitecore/Content/Product Repositories` item, click **Insert**, and then click **Insert from Template**.



- The *Brands* item appears below the repository.

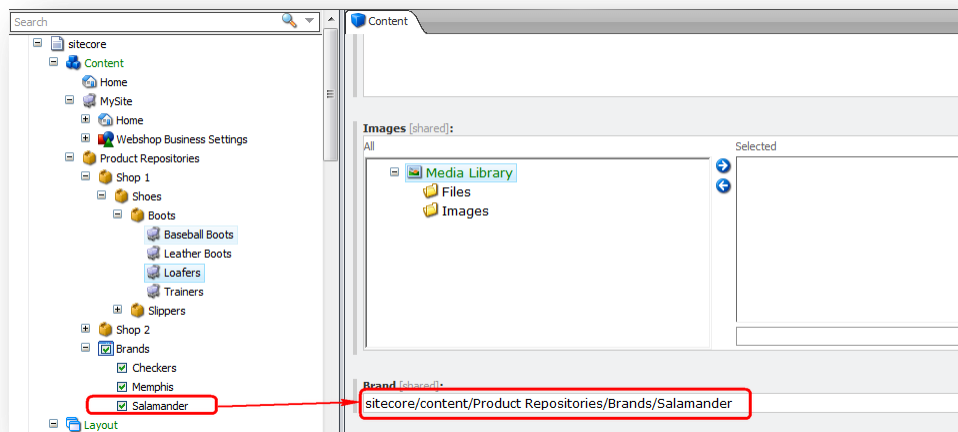


- Under the *Brands* item, create search list options that are based on the *Attribute* template.

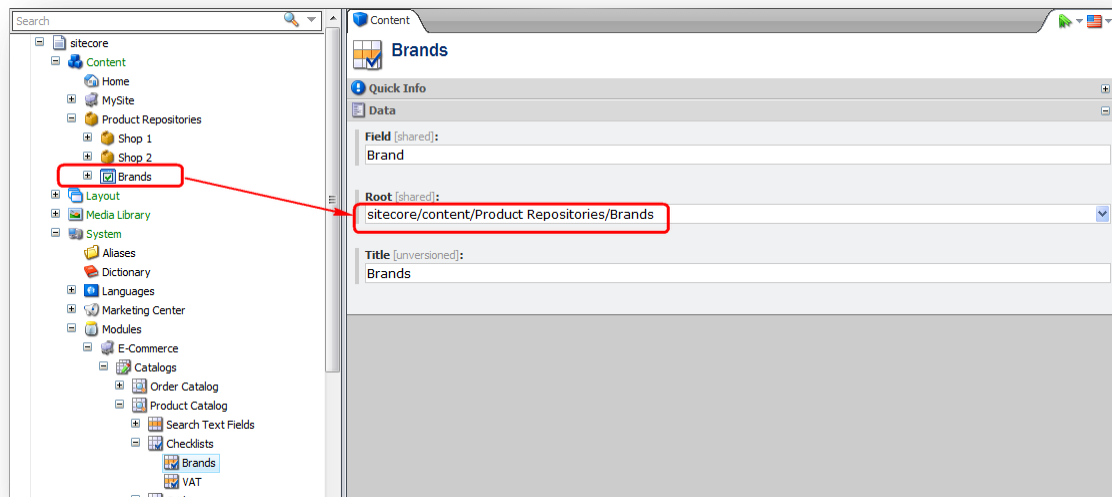


- Now you can assign some brands to the products in question. To do this, click the product that you want to assign the brand to. Scroll down to the *Brand* field, which is integrated in the *Product* template by default. Navigate to the `Sitecore/content/Product`

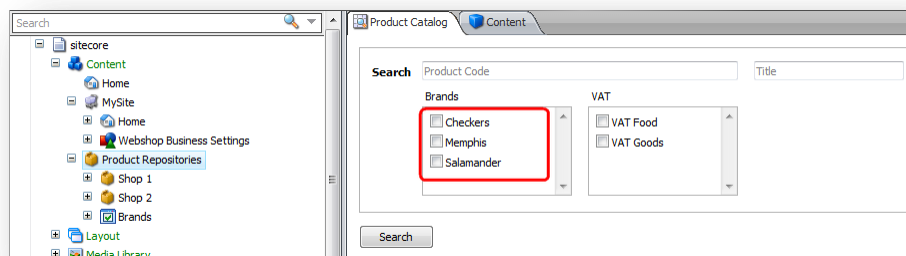
Repositories/Brands/Checkers item.



7. In the System/Modules/E-Commerce/Catalogs/Product Catalog/Checklists/Brands item, in the *Root* field, select the path to the *Brands* search options list.



8. Click any *Product Catalog* page, and you can see that it displays the search options.



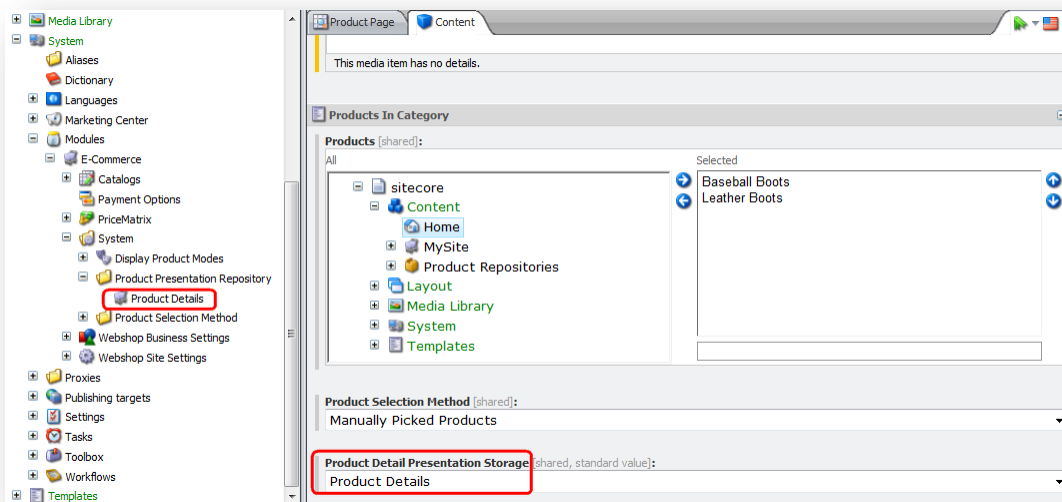
5.2 Configuring the Presentation Settings

You can find the presentation settings in the `Sitecore/System/Modules/System` folder. The presentation settings allow you to configure the way the webshop pages are presented to the webshop customers. The `Sitecore/System/Modules/System` folder contains 3 types of the presentation settings:

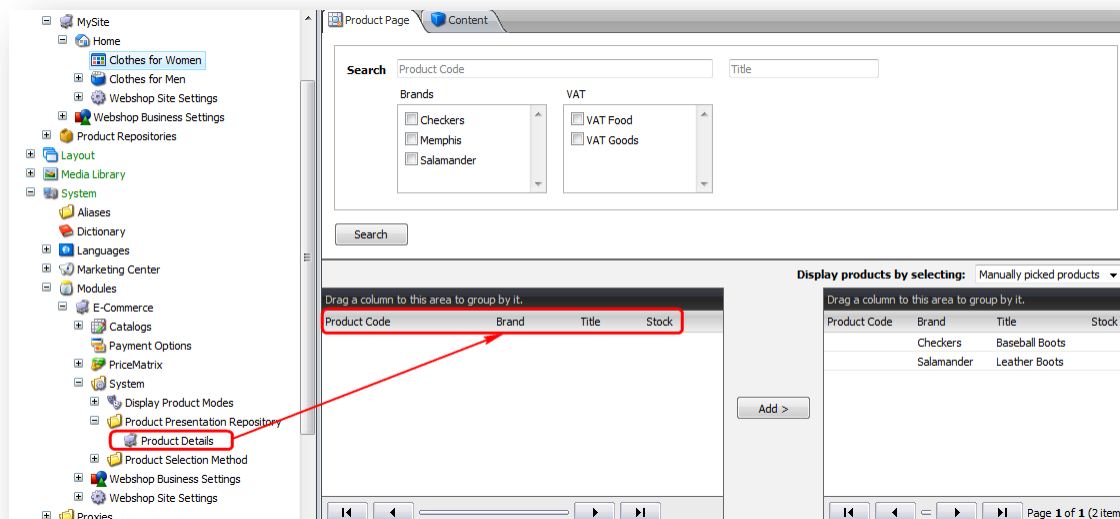
- Product Presentation
- Product Display Modes
- Product Selection Method

5.2.1 Product Details and Presentation

The *Product Details* item in the `Sitecore/System/Modules/E-Commerce/System/Product Presentation Repository` folder specifies where the template that defines the appearance or presentation of a product is stored.



In other words, it defines which columns are displayed in the search forms on the **Product Page**.

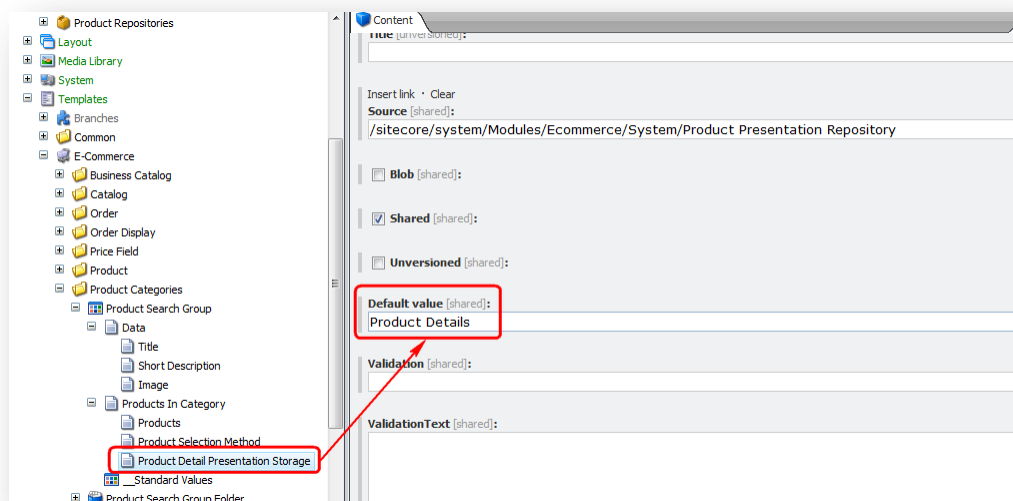


Note

The *Product Details* presentation is a setting that is stored separately from the settings of products in the **Product Repositories** and separately from the Product Catalog settings. This allows you to display the same products on multiple website pages in different ways. You can have one set of product fields displayed on one webpage and a different set of product fields displayed on another webpage.

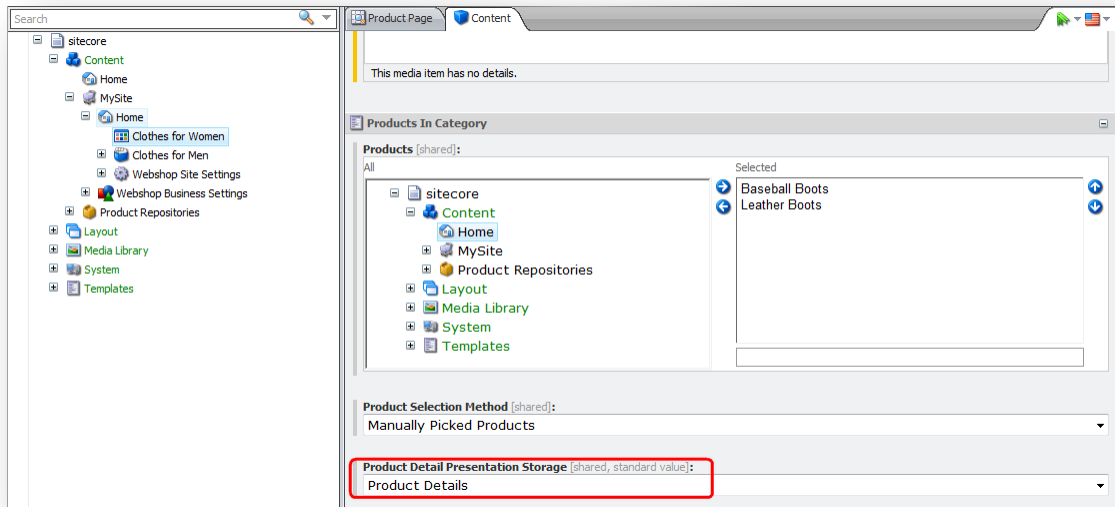
You can configure a presentation setting (in this case *Product Details*) so that it is inherited on lower level pages of your webshop. To do this:

1. Enter the default value in the *Product Detail Presentation Storage* field in the *Product Search Group* template.

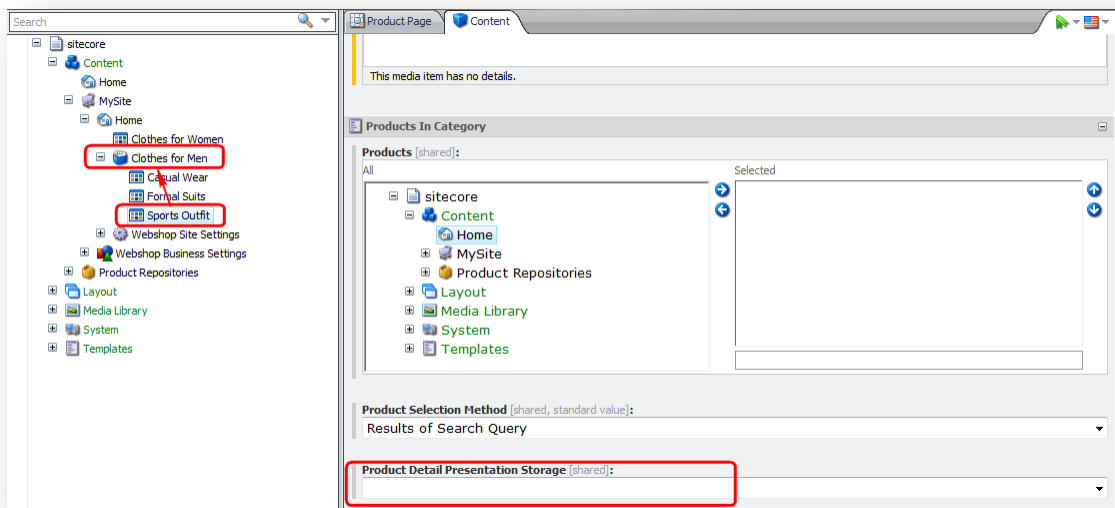


This ensures that everywhere that the *Product Search Group* template is used, the default value is selected.

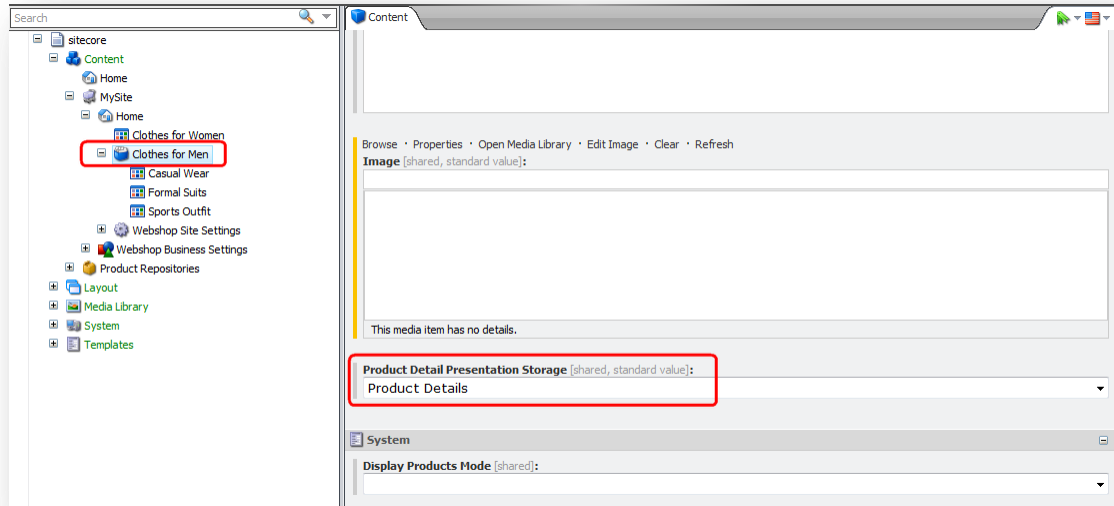
2. Enter a particular value in the **Product Detail Presentation Storage** field on the actual item and this value is used.



3. Clear the value in the **Product Detail Presentation Storage** field on the item.



The search will continue up the tree until it finds an item with the field filled in.



For more information about how to configure *Product Details*, see the *SES Configuration Guide*.

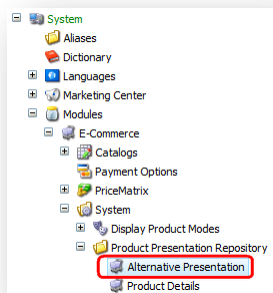
Creating the Product Details Presentation

In SES, the product information is stored separately from the presentation information for the webshop pages where these products are displayed. This is why; you can display products differently on different webshop pages.

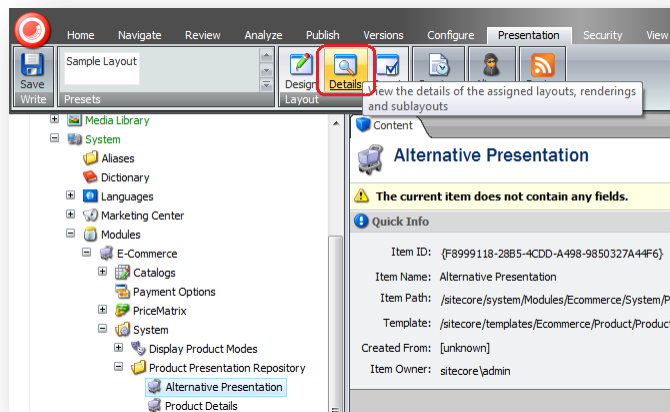
If you want to use a presentation that is different from the default presentation for your webshop pages, you need to create a new presentation.

To create a new presentation:

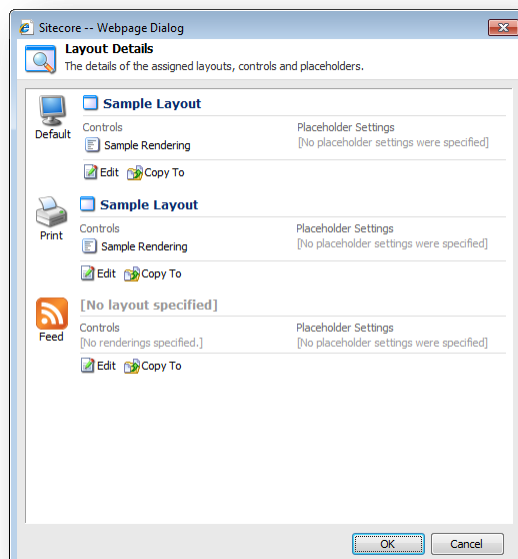
1. In the Sitecore/System/Modules/E-Commerce/System/Product Presentation Repository folder, copy and rename the *Product Details* presentation item.



2. Select the new presentation item and on the **Presentation** tab, in the **Layout** group, click **Details**.



The **Layout Details** dialog box is displayed.

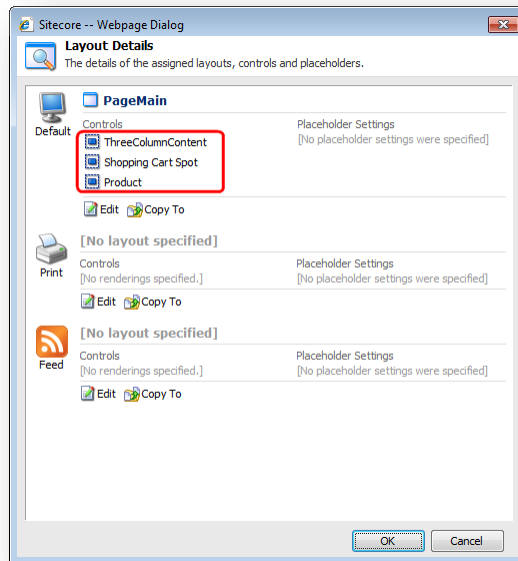


3. In the *Default* presentation, click **Edit** to add the controls that you want.

Note

To create a presentation, you must create your own sublayouts and renderings, which you can then add to the *Default* presentation. For more information about how to create layouts, see the *Presentation Component Cookbook*. For more information about presentation components, read the *Presentation Component Reference*.

4. When you add new sublayouts and renderings, the **Layout Details** dialog box lists them in the *Default* presentation:



5.2.2 Product Display Modes

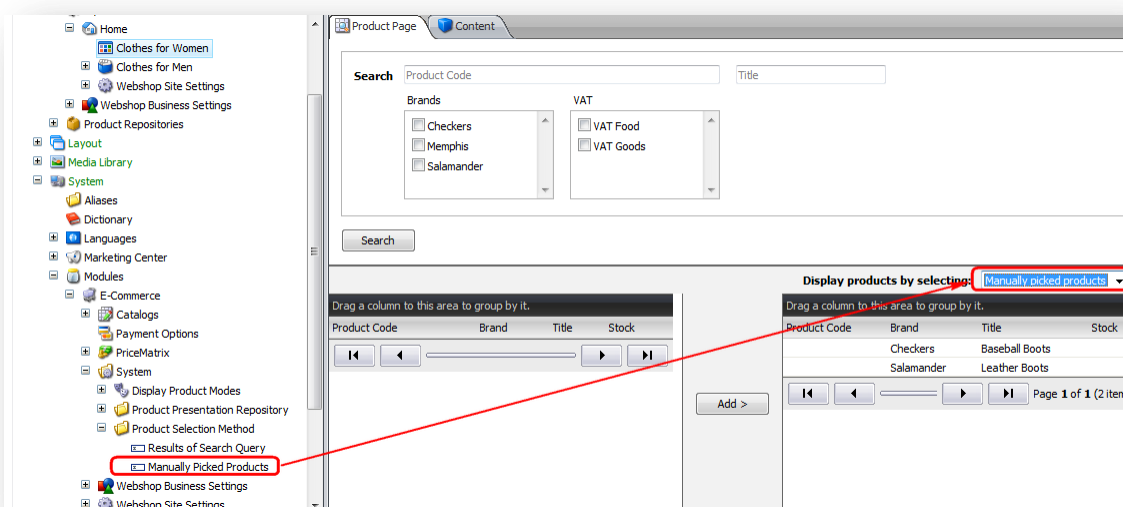
The Product Display Modes is a setting that allows you to display products in the product URL in a different way. SES supports the following URL display modes:

- Resolve Products By Item Name
- Resolve Products by Item Name and Product Code
- Resolve Products by Product Code

For more information about how to display product modes, see chapter *Display Product Modes* of the *SES Configuration Guide*.

5.2.3 Product Selection Method

The *Product Selection Method* folder contains the settings that allow you to configure how products are displayed on the product pages of your webshop.



For more information about the *Product Selection Method* folder, see the *SES Configuration Guide*.

Chapter 6

Business Settings

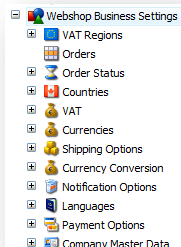
This chapter describes different types of Webshop Business Settings, how to create the Webshop Business Settings item, select the right webshop business settings, and create a Webshop Business Settings option. You will also be able to create the checkout process.

The chapter contains the following sections:

- Webshop Business Settings
- Creating Checkout Process

6.1 Webshop Business Settings

The *Webshop Business Settings* contain all of the items that affect the business aspects of your webshop, such as, the list of currencies that the webshop supports, what delivery alternatives are available, and so on.



6.1.1 Overview of Webshop Business Settings

The following table lists all of the business settings used in SES:

Webshop Business Setting	Purpose	Required or Optional
Countries	Contains all of the countries that your webshop ships orders to.	Optional
Currencies	Contains all the currencies that your webshop supports.	Optional
Currency Matrix / Conversion	Allows you to define currency rates.	Optional
Languages	Contains the languages that can be defined for mapping the languages in external ERP systems.	Optional
Notification Options	Contains the different notification options.	Required
Order State	Contains the different order states and states configuration settings.	Required
Payment Options	Contains the different payment options.	Required
Shipping Providers	Contains the different shipping options for delivering products to your customers.	Required
VAT	Contains the different VAT rates that you webshop supports. Countries have specific VAT rates that are different from the VAT rates in other countries.	Required
VAT Regions	Contains the different VAT regions that you webshop supports.	Optional

For more information about the *Webshop Business Settings*, see the *SES Configuration Guide*.

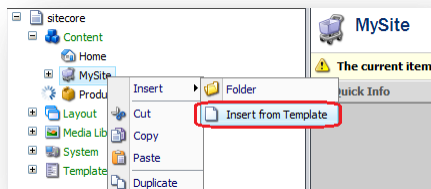
For more information about the *Orders* setting, see the *Extending Order Lines Manual*.

6.1.2 Creating the Webshop Business Settings Item

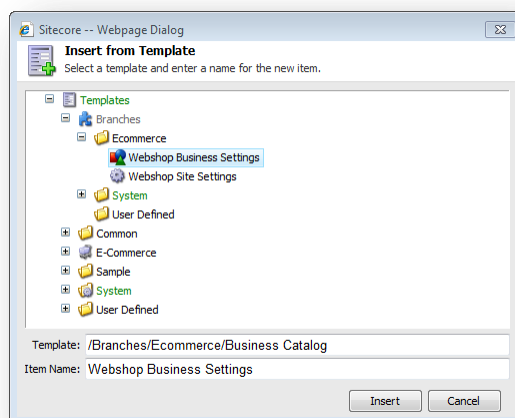
The E-Commerce core package includes the *Webshop Business Settings* template. Use the *Webshop Business Settings* template to create and configure your webshop settings.

To create the *Webshop Business Settings* item:

1. Right-click the `Sitecore/Content/MySite` item, click **Insert** and then click **Insert from Template**.

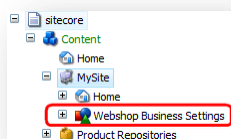


2. In the **Insert from Template** dialog box, navigate to the *Webshop Business Settings* template.

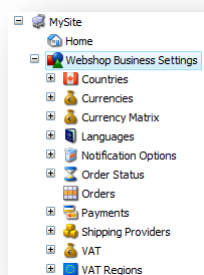


3. In the **Insert from Template** dialog box, in the **Item Name** field, enter *Webshop Business Settings* and click **Insert**.

A new *Webshop Business Settings* item appears in the *MySite* folder.



4. Expand the *Webshop Business Settings* item to see all the available webshop business settings that you can use to configure your webshop pages. For more information, see the *SES Configuration Guide*.



6.1.3 Selecting the Right Webshop Business Settings

You must decide which webshop business settings you need for your webshop. You can create and add other settings to the existing Webshop Business Settings. For more information about how to create or edit a webshop business setting, see the chapter *Creating a Webshop Business Setting Option*.

You will probably use the following webshop business settings for your webshop:

- Notification Options
- Order State
- Payment Options
- Shipping Providers
- VAT

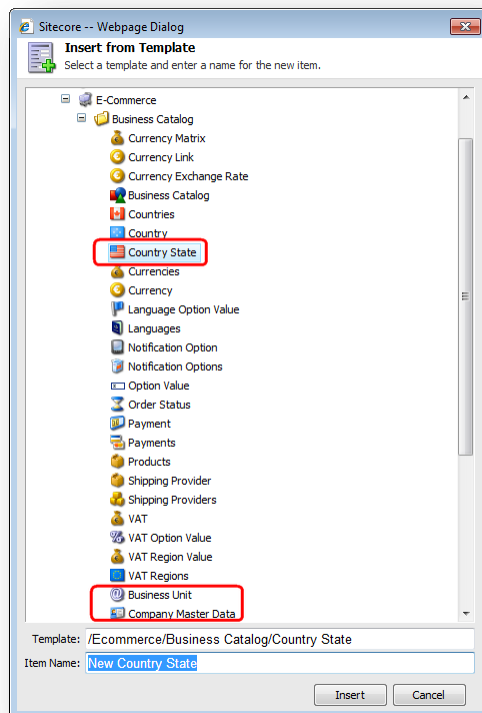
You may consider using some other webshop business settings, if your webshop ships goods abroad:

- Countries
- Currencies
- Currency Matrix
- Languages
- VAT Regions

You can create the following webshop business settings from the existing templates:

- Country State
- Business Unit

- Company Master Data



You can create custom webshop business settings based on the existing Business Catalog templates or based on other Sitecore templates.

The webshop business settings can be used on the following pages of the checkout process:

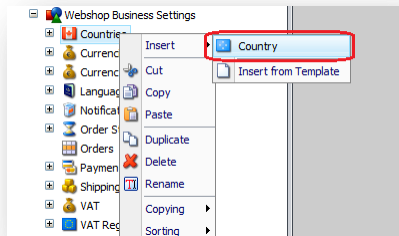
- Shopping Cart
 - Currencies
 - VAT
 - VAT Regions
- Payment
 - Notification options
 - Payments
 - Shipping Options
- Confirmation
 - Order State
 - VAT
 - Notification options
 - Shipping Options

6.1.4 Creating a Webshop Business Setting Option

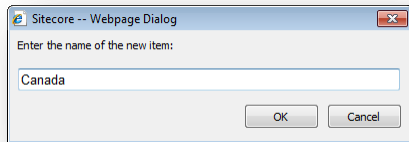
You can create your own *Webshop Business Setting* option.

To create a *Webshop Business Setting* option:

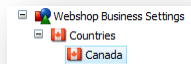
1. Right-click the setting to which you want to add an option, click **Insert**, and then click **Country**.



2. The dialog box appears where you must enter the name of the country.

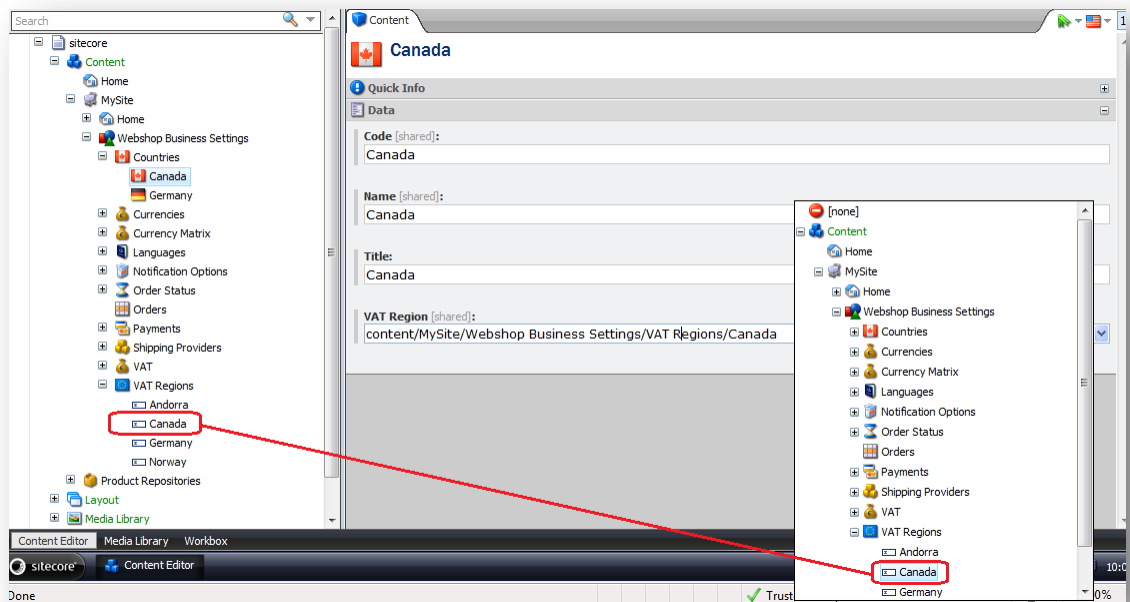


3. Enter the name of the country, click **OK** and the option appears under the **Countries** setting.

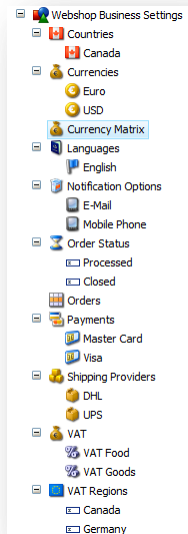


You can change its icon to match the country name.

4. You can link an option setting to another *Webshop Business Setting*.



5. Create the options for the settings that you need.



For more information about how these setting options work, and what their fields mean, see the *SES Configuration Guide*.

6.1.5 Payment Options

You must use the Payment Options setting to configure the different methods of payment that you want your webshop to support.

SES supports two payment methods by default:

- Checking Account
- Money Transfer

However, Sitecore also distributes a number of packages that contain payment providers. These providers allow you to configure a number of different online payment services and integrate them into your webshop. These payment provider packages contain the code and the item you need to configure the corresponding payment option.

You can install as many of these packages as you need in your SES solution. You can download these packages from the [SDN](#) and install them individually on your SES installation.

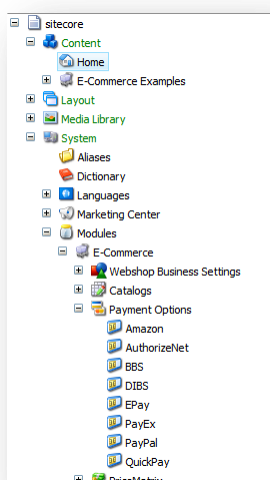
All the payment providers support:

- Money reservation and ticket booking.
- Ticket cancellation and payment capturing in the Order Catalog.

The payment providers are not part of the SES core package or of the *Sitecore E-Commerce Example Pages* package.

Configuring a Payment Option

When you install a payment provider package, an item is created in the `/sitecore/system/Modules/Ecommerce/Payment Options` folder.



To configure a payment provider, you must copy the corresponding new payment option item from here and place it in the *Webshop Business Settings* folder for your webshop, for example, `MyWebshop/Home/Webshop Business settings/Payment Options`.

When you install a payment provider package, most of the fields in the payment option item are already filled in. However, you *must* edit the **User Name** (Merchant ID) and **Password** fields and ensure that they contain the relevant information about the payment service that you want to use.

You must copy the payment option items that you want to use from the `E-Commerce Examples/Home/Webshop Business settings/Payment Options` folder and place them in the *Webshop Business Settings* folder for each webshop installation that you want to run.

For more information about the how install and configure a payment provider package as well as information about the Payment Provider API, see the *SES Payment Provider Guide*.

6.2 Creating Checkout Process

You can include the following pages in the checkout process:

- Shopping Cart
- Customer Details
 - Login Credentials
 - Billing Address
 - Shipping / Delivery Address
- Payment
 - Payment Return Page
 - Cancelled Payment Page
 - Payment Error
- Payment Confirmation
 - Login Credentials
 - Billing Address
 - Shipping / Delivery Address
 - Order Details
- Order History
 - Order Details

Of course, you can include some or all of these pages. You can split or merge pages and add new pages according to your requirements and specifications.

For more information about how to build a checkout process, see the *SES Configuration Guide*.