

# Kentico CMS 7.0 Tutorial ASPX

# Table of Contents

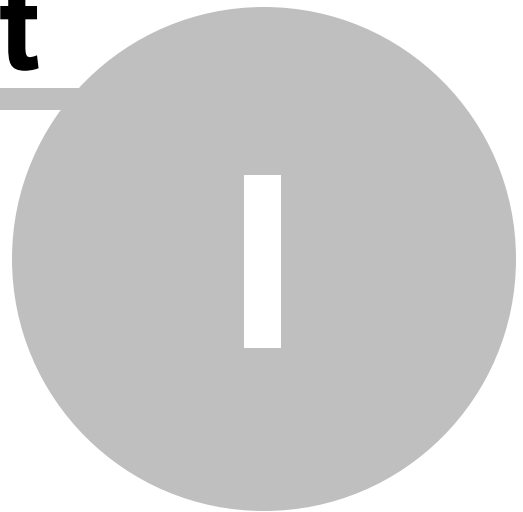
<b>Introduction</b>	<b>5</b>
Kentico CMS Overview.....	5
<b>Installation</b>	<b>7</b>
Prerequisites.....	7
Setup installation.....	8
Web application installation.....	9
Database setup and Corporate Site.....	11
<b>Managing content</b>	<b>17</b>
User interface overview.....	17
Editing home page content.....	20
Creating a new page.....	21
Inserting an image.....	25
Creating a link.....	26
Creating a news item.....	28
<b>Site development overview</b>	<b>32</b>
Site development overview.....	32
<b>Creating pages using ASPX templates</b>	<b>37</b>
ASPX page templates.....	37
Creating a simple ASPX page template.....	39
Using master pages.....	45
Adding portal engine functionality to ASPX templates.....	48
<b>Managing styles and design</b>	<b>56</b>
CSS styles.....	56
App themes.....	58
Menu design.....	59
<b>Creating a new site using ASPX templates</b>	<b>63</b>
Overview.....	63
Creating a new web site using the wizard.....	63
Creating the CSS stylesheet.....	68
Opening and configuring the web project.....	71
Master page.....	72

---

<b>Main menu</b> .....	<b>77</b>
<b>Home page</b> .....	<b>78</b>
<b>News page</b> .....	<b>87</b>
<b>Services page</b> .....	<b>92</b>
<b>Products page</b> .....	<b>97</b>
<b>Overview</b> .....	<b>97</b>
<b>New document type</b> .....	<b>97</b>
<b>Transformations</b> .....	<b>103</b>
<b>Page template</b> .....	<b>106</b>
<b>Search page</b> .....	<b>110</b>
<b>Secured section for partners</b> .....	<b>115</b>
<b>Further steps</b> .....	<b>123</b>
<b>Further steps</b> .....	<b>123</b>

**Part**

---



**Introduction**

---

# 1 Introduction

## 1.1 Kentico CMS Overview

Kentico CMS for ASP.NET helps you create powerful dynamic websites with minimum effort. This document will guide you through the most important features of the system step-by-step, so that you can start creating your own websites.

This document was written for evaluators and new users. It's intended for developers who create the websites. It's not intended for end-users without programming knowledge.

If you need a more detailed documentation of some features, please see one of the following documents:

- Developer's Guide
- Controls Reference
- Web parts or Widgets Reference
- API Reference
- Database Reference

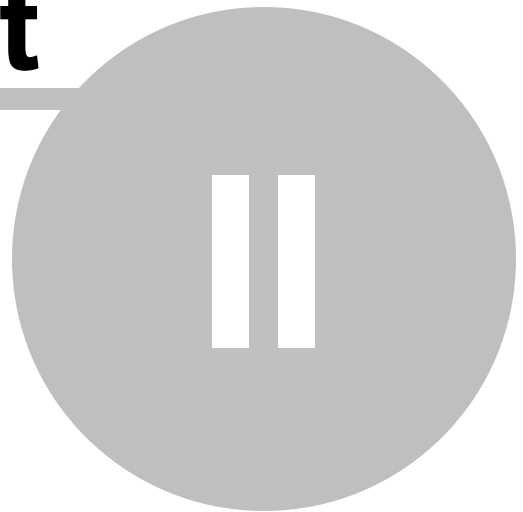


### **Kentico CMS Support**

You get free technical support during your evaluation period. If you need any help, please visit <http://www.kentico.com/Support.aspx>.

# Part

---



## Installation

---

## 2 Installation

### 2.1 Prerequisites

The following configurations are supported by Kentico CMS. Other configurations may work too, but the system was not tested on them.

#### Server-side Requirements

- Windows Vista Home Premium/Business/Enterprise/Ultimate, Windows 7 (both 32bit and 64bit) or Windows 8 (both 32bit and 64bit), or Windows Server 2008, 2008 R2, 2012.
- Microsoft .NET Framework [3.5 SP1](#) or [4.0](#) or higher.
- Microsoft Internet Information Services (see the table below) or Visual Studio/Visual Web Developer 2008/2010/2012 built-in web server.
- Microsoft SQL Server 2005, 2008, 2008 R2, 2012 (including free SQL Server Express Edition [2005/2008/2012](#)).

#### Internet Information Services overview

Internet Information Services version	Operating system	Details and installation instructions
IIS 5.1	Windows XP Professional	not supported
IIS 6.0	Windows Server 2003	not supported
IIS 7.0	Windows Vista	<a href="#">IIS 7 Installation and Deployment</a>
	Windows Server 2008	
IIS 7.5	Windows 7	
	Windows Server 2008 R2	
IIS 8.0	Windows 8	<a href="#">Installing IIS on Windows Server 2012</a>
	Windows Server 2012	

#### Hosting Requirements

- ASP.NET 3.5 SP1 (or higher) and Microsoft SQL Server 2005/2008/2012 support.
- Medium-trust or full-trust permissions for the ASP.NET application.
- If the server uses medium trust, ASP.NET AJAX 1.0 must be installed on the server.
- If the application uses .NET Framework 3.5 SP1 and is hosted in a medium trust environment, it is necessary to have [Microsoft Chart Controls](#) installed on the server.
- It's recommended that your hosting plan comes with 250 MB or more memory and 100+ MB database.

You can use your favorite hosting provider or choose from our [hosting partners](#).

## Windows Azure

Kentico CMS fully supports the Microsoft Windows Azure platform, including SQL Azure, Azure Storage and other services. Windows Azure SDK 1.7 is required.

## Development Tools

If you want to create custom web parts or integrate custom code, you need **Visual Studio 2008/2010/2012** or free **Visual Web Developer 2008/2010/2012 Express Edition**.

## Supported Client Browsers for Content Editors

- Internet Explorer 8, 9
- Firefox 4.0+
- Chrome 12
- Safari 4.0+ or Firefox 4.0+ on Mac OS

## Supported Client Browsers for Site Visitors

- Internet Explorer 6.0+
- Firefox 1.0.5+
- Chrome 12+
- Mozilla 1.7.1+
- Netscape 7.1+
- Opera 7.52+
- Safari or Firefox on Mac OS
- Mobile browsers, such as Safari on iPhone, are supported as well, but some features may be limited by browser capabilities.

(the visitor browser requirements also depend on the functionality used on the website)

## Required experience

Although Kentico CMS allows you to create dynamic websites without programming, you may want to create custom web parts or add custom code when developing more complex websites. Developers should be able to create a simple ASP.NET application using Visual Studio and have some experience with relational databases and SQL, so that the flexibility of Kentico CMS can be fully put to use.

## 2.2 Setup installation

### Troubleshooting installation issues

If you encounter any problems during the installation, please see **Kentico CMS Developer's Guide -> Installation and deployment -> Troubleshooting installation issues** or contact our support at <http://www.kentico.com/Support.aspx>



Run KenticoCMS.exe and follow the installation wizard:



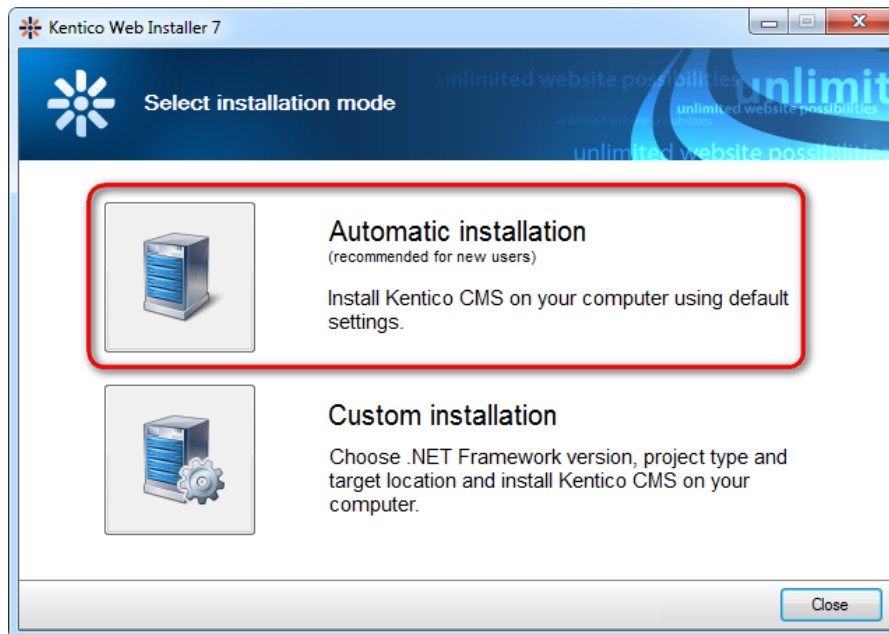
Read and accept the license agreement and click **Next**. Choose the installation location of the binary files and documentation on your disk. After the setup completes the installation, check **Launch Kentico Web Installer** and click **Finish**.

## 2.3 Web application installation

The Kentico Web installer opens. If it has not started automatically, you can always run it from **Start menu -> All Programs -> Kentico CMS -> Kentico Web Installer**.

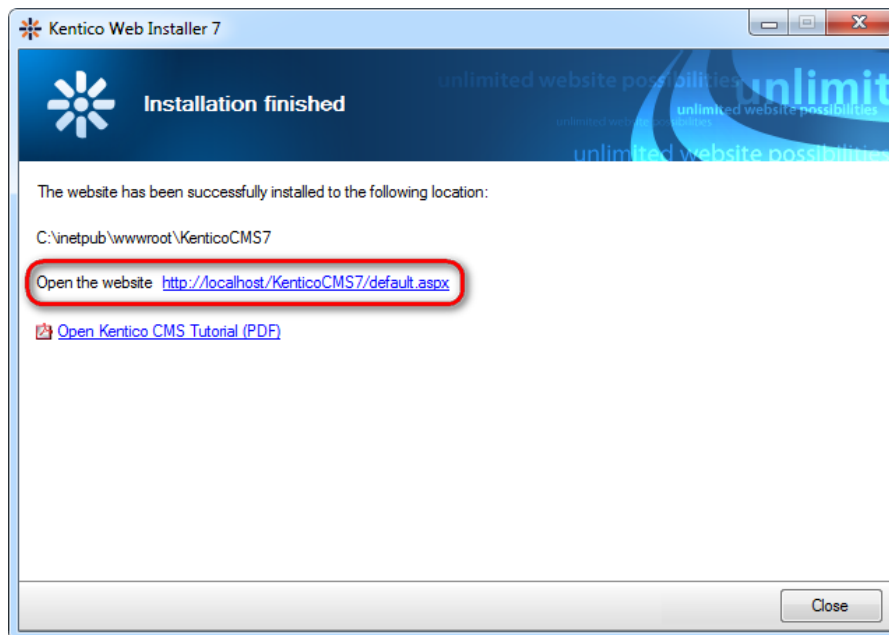
### Selecting installation mode

1. In the installer, select the **Automatic installation** option.



Note that **Automatic installation** chooses .NET framework based on the framework that is available in your system, first checking for .NET 4.0, .NET 4.5 and finally 3.5. *Web site project* and default IIS *inetpub\wwwroot* installation folder are selected. If you want to make these selections yourself, refer to [Developer's guide -> Installation and deployment -> Installation procedure -> Web installer](#) for more information on the **Custom installation** option.

2. Once the installer copies the necessary files, click the **Open the website** link.



3. Continue to the [Database setup](#) topic.

## 2.4 Database setup and Corporate Site

Now you should see the **Database setup** in your web browser.

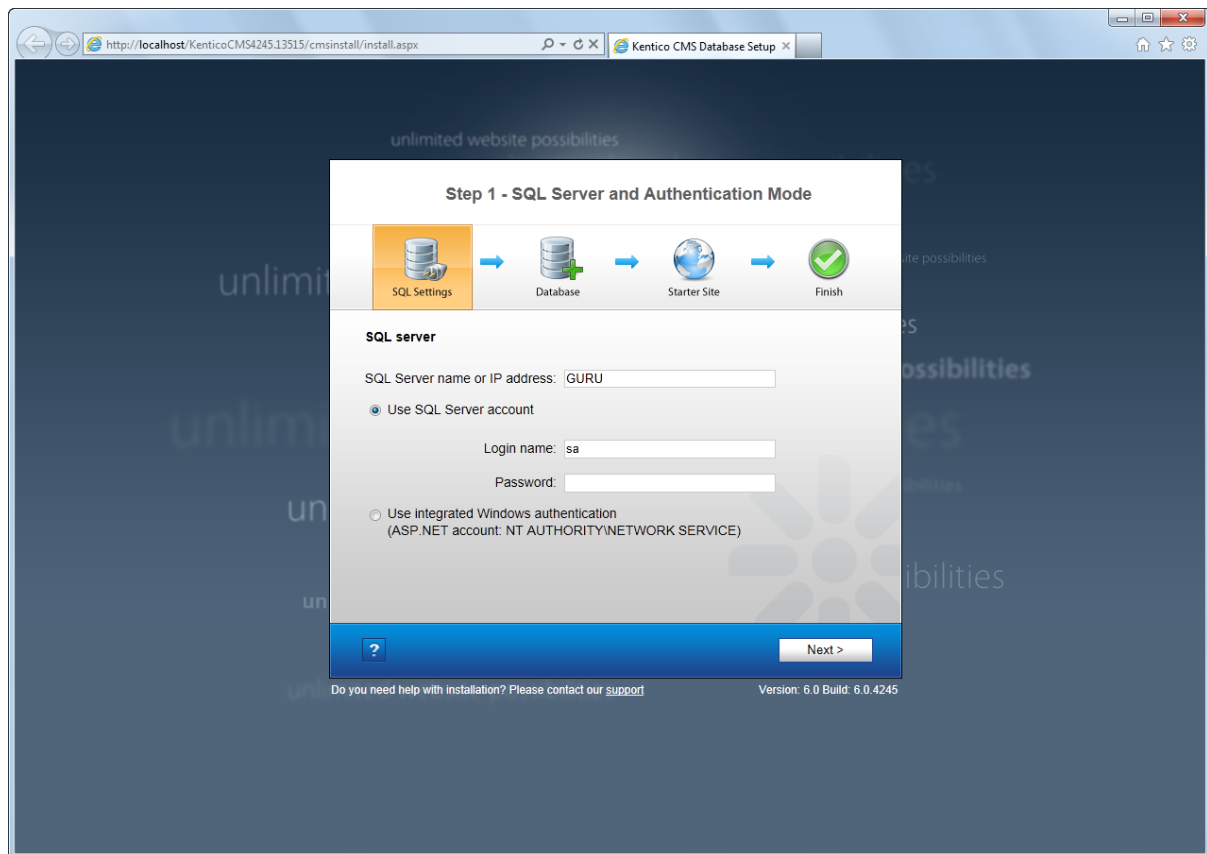
In the first step, choose the SQL Server name or IP address. If you are using SQL Server 2005 Express Edition, the default server name is `.\SQLExpress` or `(local)\SQLExpress`.

You can use either SQL Server authentication (recommended) or integrated Windows authentication.

- In case you use **SQL Server account**, you need to enter the user name (such as **sa**) and password.
- In case you use **Windows authentication**, you need to ensure that the ASP.NET account of the name displayed in the brackets has an appropriate login name in your SQL Server.

Permissions for creating a new databases or for creating database objects in an existing database must be granted to the account.

Click **Next**.




The screenshot shows a web browser window displaying the 'Kentico CMS Database Setup' wizard. The title bar indicates the URL is `http://localhost/KenticoCMS4245.13515/cmsinstall/install.aspx`. The main content area features a progress bar with four steps: 'SQL Settings', 'Database', 'Starter Site', and 'Finish'. The 'SQL Settings' step is currently active. Below the progress bar, the 'SQL server' section contains the following fields and options:

- SQL Server name or IP address:
- Use SQL Server account
  - Login name:
  - Password:
- Use integrated Windows authentication (ASP.NET account: NT AUTHORITY\NETWORK SERVICE)

At the bottom of the form, there is a blue bar with a help icon on the left and a 'Next >' button on the right. Below the form, a footer line reads: 'Do you need help with installation? Please contact our [support](#) Version: 6.0 Build: 6.0.4245'.

In the **Database Instance** step, choose **Create a new database**, enter the name of the new database into the **New database name** field and click **Next**.

### Step 2 - Database Instance



**Database**

Create a new database

New database name:

Use an existing database

Existing database name:


Create Kentico CMS database objects

[Show advanced options](#)

[?](#)

The database creation log will be displayed.

### Step 3 - Database Creation Log



Creating database objects...

```
proc_forums_forum_setpermissions
proc_forums_forum_removedependencies
proc_forums_forum_movenodeup
proc_forums_forum_movenodedown
proc_forums_forum_initnodeorders
proc_forums_forumpost_updatethreadcounts
proc_forums_forumpost_updatepostcounts
proc_forums_forumpost_unstickthread
proc_forums_forumpost_stickthread
proc_forums_forumpost_removedependencies
proc_forums_forumpost_moveupstickythread
proc_forums_forumpost_movedownstickythread
proc_forums_forumgroup_removedependencies
proc_forums_forumgroup_movenodeup
proc_forums_forumgroup_movenodedown
```

When the database is created, you will be asked to enter your license key. If you do not have a license

key yet, click **Next** to continue in trial mode. The functionality of the trial mode is the same as the full version.

### Step 4 - License Key



SQL Settings → Database → Starter Site → Finish

Your trial license key has expired, please enter a valid license key for domain: localhost

**Tip:** If you need a temporary license key for this domain, please write to [support@kentico.com](mailto:support@kentico.com) and ask for a trial key for the following domain name: localhost

You can also get a license of Kentico CMS Free Edition after registration at <http://www.kentico.com>.

Please enter the key:

[Skip this dialog](#)

?Next >

In the **Starter Site** step, you can choose from the following options:

- **Choose a starter site:**
  - **Corporate Site (portal engine)** - this option installs the sample corporate site. This is **recommended** for most users, especially **for evaluators**.
  - **E-commerce Site** - this sample site can be used as a starting point for creating your own e-shop and shows the possibilities of Kentico CMS's E-commerce module.
  - **Personal Site** - this is a web template suitable for a simple personal site.
  - **Community Site** - complex web template suitable for community webs, showing Kentico CMS's social networking features in practice.
  - **Intranet Portal** - ready-to-use solution for company intranets with support of departments, workgroups, project management, etc.
  - **Blank Site** - this is a blank site without any content; you will use it to create a new site from scratch.
  - **Blank Site ASPX** - the same as above, but for ASPX page templates.
- **Continue to the New site wizard** - this option is recommended if you are starting a new site from scratch.
- **Import an existing Kentico CMS website** - use this option if you already created a website with Kentico CMS and need to import it into the new installation (e.g. on the production server).

For the purposes of this guide, please select the sample **Corporate Site** and click **Next**. You will see the confirmation and a link to your new website:

### Step 5 - Starter Site



SQL Settings → Database → Starter Site → Finish

Choose starter site

**Corporate Site**

This is a web template for a general corporate site. It's used as a showcase of Kentico CMS capabilities and it can be used as a starting site that you modify as needed. It uses the portal engine and it's the recommended choice for developers who are new to Kentico CMS.

**E-commerce Site**

This is a web template for a simple E-commerce site. It's used as a showcase of Kentico CMS E-commerce module capabilities and it can be used as a starting site that you modify as needed. It uses the portal engine and it's the recommended choice for developers who are new to Kentico CMS.


Continue to the New site wizard

Import existing Kentico CMS website

[?](#) [Next >](#)

A log showing the creation of the site will be displayed. When it's finished, you will be shown the final step that you can see in the screenshot below. Click on **Continue to the new web site**.

### Step 7 - Finished

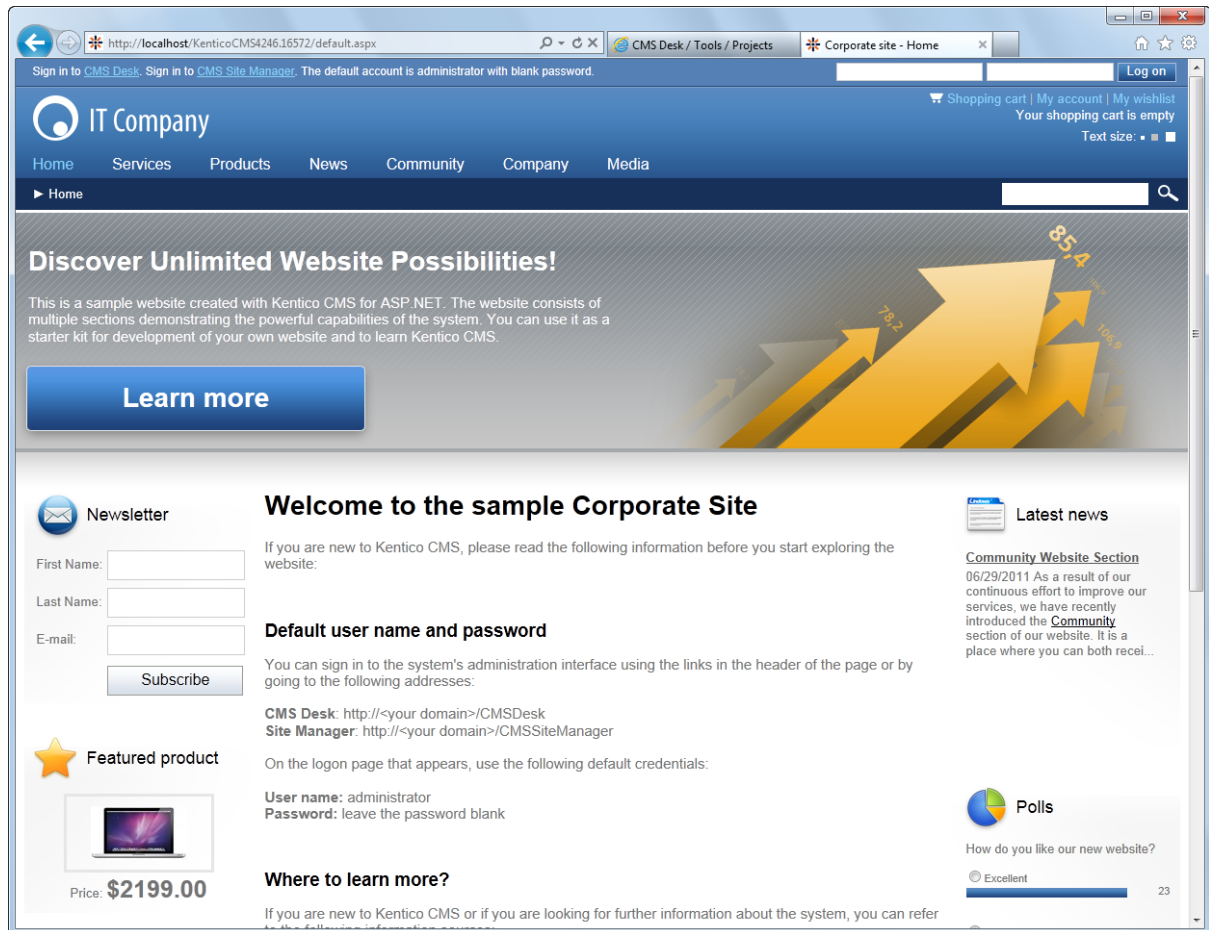


SQL Settings → Database → Starter Site → Finish

The site has been created successfully.

[Continue to the new website](#)

You will be redirected to the title page of the sample Corporate Site:



The screenshot shows a web browser window displaying the sample Corporate Site. The browser address bar shows the URL: `http://localhost/KenticoCMS4246.16572/default.aspx`. The page header includes the site logo "IT Company" and navigation links: Home, Services, Products, News, Community, Company, Media. A shopping cart icon indicates "Your shopping cart is empty".

The main content area features a large banner with the heading "Discover Unlimited Website Possibilities!" and a "Learn more" button. Below the banner, there are several sections:

- Newsletter:** A form with fields for "First Name", "Last Name", and "E-mail", and a "Subscribe" button.
- Welcome to the sample Corporate Site:** A section providing information for new users, including default user name and password, and links to CMS Desk and Site Manager.
- Latest news:** A section titled "Community Website Section" dated 06/29/2011.
- Featured product:** A section showing a product image and a price of "\$2199.00".
- Where to learn more?:** A section providing information for new users or those seeking further information.
- Polls:** A section titled "How do you like our new website?" with a poll showing "Excellent" as the selected option.

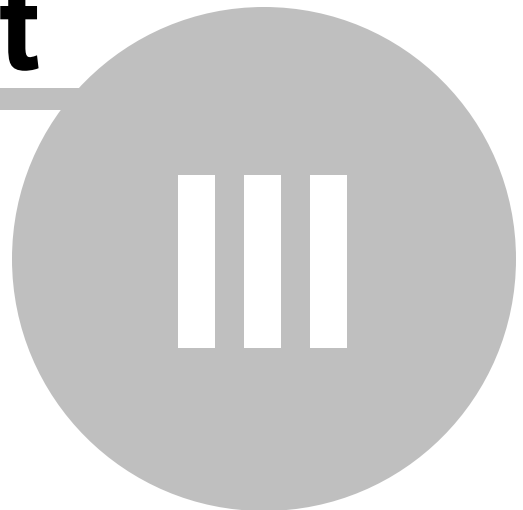


### Sample website

The Corporate Site website is only an example of a website you can create with Kentico CMS. You have full control over the site structure, design, page layout and functionality as you will see in the following chapters.

# Part

---



**Managing content**

---



## 3 Managing content

### 3.1 User interface overview

Click **Sign in to CMS Desk** at the top of the website or go to <http://<domain>/<virtualdirectory>/cmsdesk>. You will be asked for a user name and password.

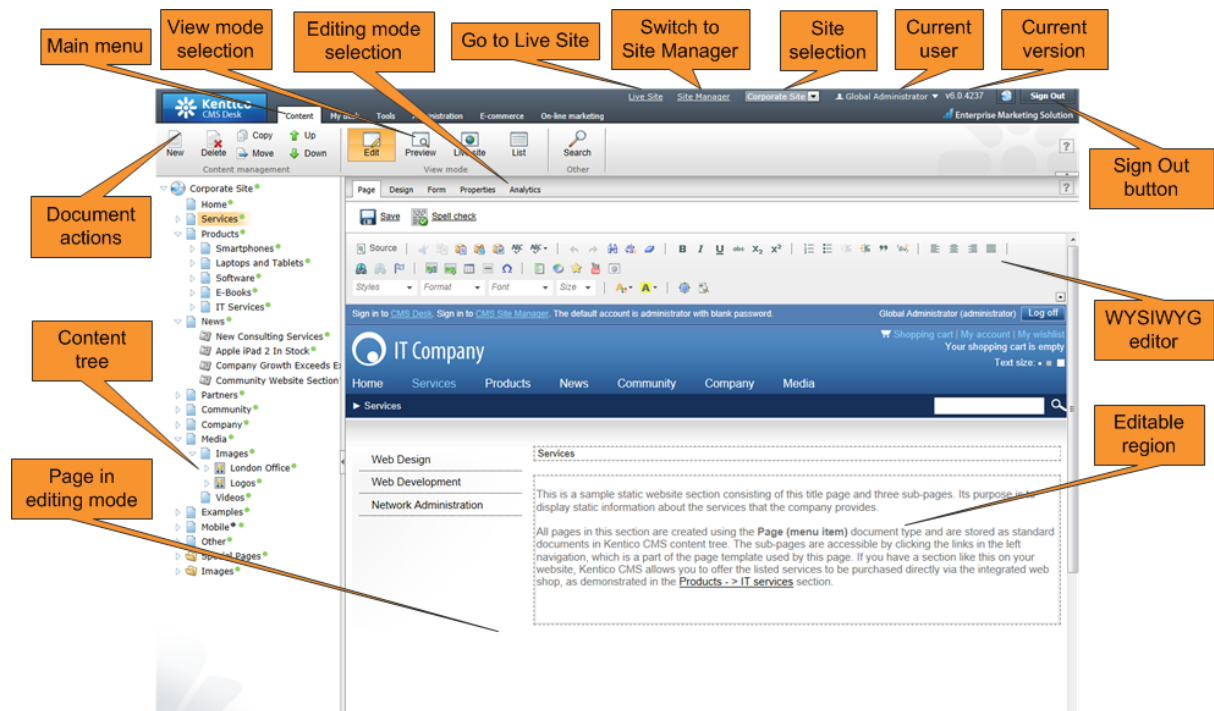


#### Default user name and password

The default user name is **administrator**, the default password is **blank (no password)**.

It is highly recommended to change the password before publishing the website on a live server.

Once you sign in, you will see a splash screen, giving you some basic information. Click the **Continue** button and you will be redirected to the following page:



The CMS Desk user interface consists of the following main sections and features:

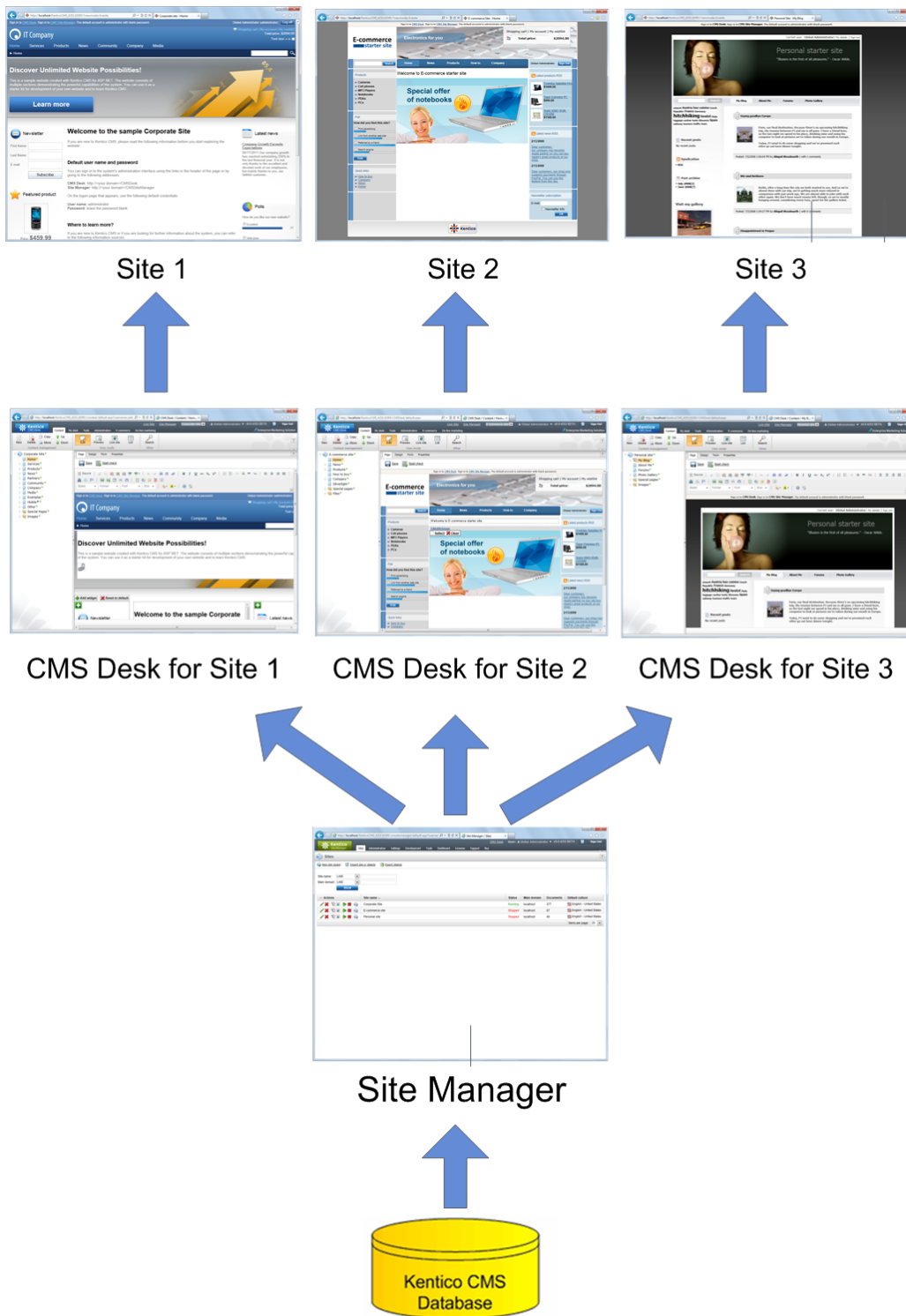
- **Main menu** with Content, My Desk, Tools, Administration, E-commerce and On-line marketing tabs.
- **Content tree** that represents the site map of the website. It allows you to organize the site's structure or select a specific document for editing in the content section.
- **Document actions** toolbar with buttons for creating new documents, deleting, copying, moving and sorting documents.

- **View mode selection** - allows you to choose between editing, preview, live site view and list view.
- **Editing mode selection** - you can choose to edit page content, design the page template, edit the document fields, product properties or document properties.
- **Live Site** - this action redirects you to the title page of the currently edited website, logged under the same user account that you used to log into CMS Desk. This is a more convenient way than using the **Sign out** button and logging in on the live site afterwards.
- **Site Manager** - redirects you to Site Manager, the other part of the system's administration interface. This option is only available for global administrators.
- **Site selection** - this drop-down list is used to select the currently edited website. Only those websites that the current user can edit are available in the list.
- **Current user** - displays the name of the current user.
- **Current version** - version number of the Kentico CMS installation.
- **Sign Out button** - clicking this button logs you out of the user interface and redirects you to the title page of the live site. This button is only displayed if *Forms authentication* is used. When using *Windows authentication*, the link is not displayed.
- **WYSIWYG editor** - allows you to edit text, change its formatting and insert graphics or other items. It is available for **Editable regions** on the *Page* tab, as well as when editing document fields on the *Form* tab.
- **Page in editing mode** - this is where you can view and edit the document selected in the content tree, in the mode selected in the view mode and editing mode toolbar.

## CMS Desk and Site Manager

**CMS Desk** allows content editors to manage the content of a specific website. Developers and site administrators who need to manage global settings, code and components that affect all websites can also use the **Site Manager** interface.

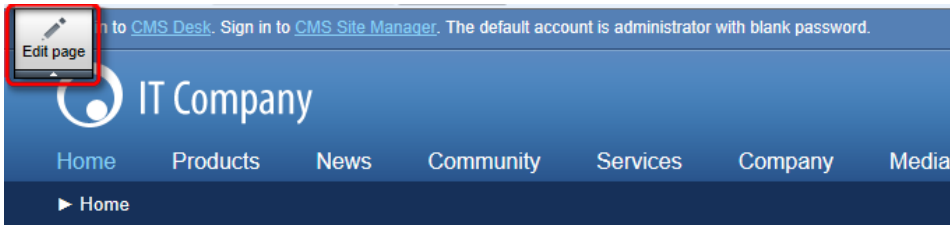
The Site Manager interface is accessible either through the **<http://<domain>/<virtualdirectory>/cmssitemanager>** URL, by clicking **Site Manager** at the top of the CMS Desk user interface or **Sign in to Site Manager** at the top of a live site. The following figure shows how the database, Site Manager, CMS Desk and websites are related:



### On-site editing

In addition to the two main administration interfaces, Kentico CMS also provides a way to edit page content directly while browsing on the live website. Authorized editors can access on-site editing mode by going to the <http://<domain>/<virtualdirectory>/cmsedit> URL, or by clicking the **Edit page** button

displayed in the corner of pages.



All actions available in on-site editing mode can also be done through the CMS Desk interface. It simply provides an alternative way to edit websites.

This tutorial demonstrates all operations in CMS Desk — it is recommended to first become familiar with CMS Desk and the general website content structure before using on-site editing. If you are interested in learning more about on-site editing, please see the [Content management -> On-site editing](#) chapter of the Kentico CMS Developer's Guide.

## 3.2 Editing home page content

Now we will modify the home page content. Click the **Home** document in the content tree. You will see a page as shown below.


A screenshot of the Kentico CMS Desk interface. The top navigation bar includes "Content", "My desk", "Tools", "Administration", "E-commerce", and "On-line marketing". The left sidebar shows a content tree with "Corporate site" expanded, and "Home" selected. The main editing area shows a rich text editor with a toolbar and a preview of the home page content. The preview includes a navigation menu, a heading "Discover Unlimited Website Possibilities!", a paragraph of text, a "Featured product" widget with a smartphone image and price "\$529.98", a "Welcome to the sample Corporate Site" section with introductory text and a "Default user name and password" section, and a "Latest news" widget with a "New Consulting Services" article dated 06/05/2011.

The page is now displayed in editing mode and it contains two editable regions. Editable regions on the **Page** tab can be identified by a dotted outline. Delete all content from one of the regions and enter the following text instead:

This is my first text.

You can then use the WYSIWYG editor toolbar at the top of the page to change the formatting of the text like this:

*This is my **first** text.*

Click the  **Save** button at the top of the page or press **CTRL+S** to save the changes.

Now click the **Live site** button on the main toolbar. You will see the modified version of the home page as it's displayed to the site visitors.



#### **Preview mode**


























If you select **Preview** mode now, it will display the same content as the **Live site** mode. It works as a preview for documents that are affected by workflow or are scheduled to be published in the future. In these cases, you can preview the latest modifications before they are published.

### 3.3 Creating a new page

Now we will create a new page under the Services section. Click **Edit** in the main toolbar to switch back to the editing mode. Click **Services** in the content tree. Click **New** in the main toolbar. You will see the following dialog that allows you to select the type of document you want to create under the currently selected document:

 **New document**

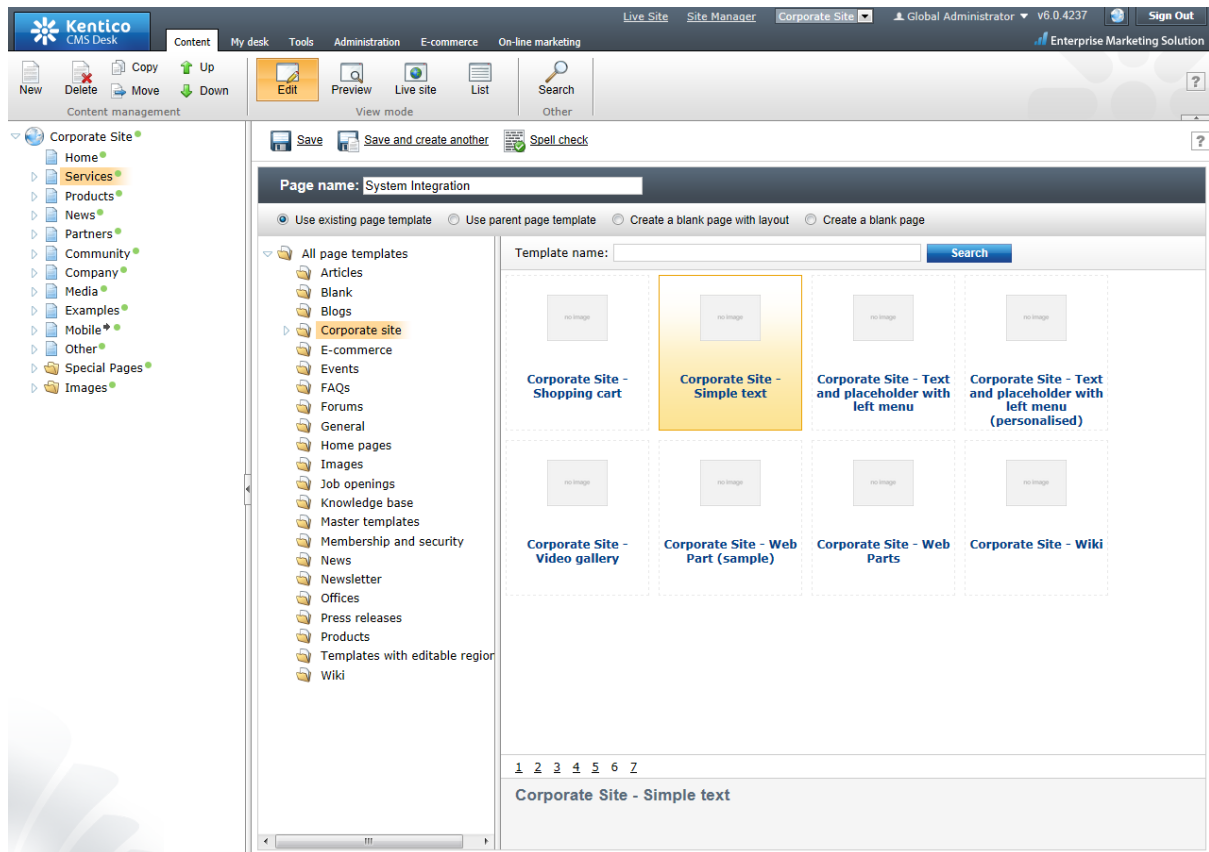
Please select new document type:



-  [Page \(menu item\)](#)
-  [Wireframe](#)
  
-  [Article](#)
-  [Blog](#)
-  [Bundle](#)
-  [Cell phone](#)
-  [E-book](#)
-  [Event](#)
-  [Event \(booking system\)](#)
-  [FAQ](#)
-  [File](#)
-  [Folder](#)
-  [Image gallery](#)
-  [IT Service](#)
-  [Job opening](#)
-  [Knowledge base article](#)
-  [Laptop](#)
-  [News](#)
-  [Office](#)
-  [Paid membership](#)
-  [Press release](#)
-  [Product](#)
-  [Simple article](#)
-  [Smartphone](#)
-  [Software](#)

---

 [Link an existing document](#)

Click the **Page (menu item)** option. You will be redirected to the new page properties dialog. Enter *System Integration* in the **Page name** field and choose the **Corporate Site -> Corporate Site - Simple text** template:



Click  **Save** to create the new page. The page is now created in the content tree and you can edit page content on the right. Enter some text in the editable regions and click  **Save** again.

The screenshot displays the Kentico CMS 7.0 interface. On the left, the content tree shows the following structure:

- Corporate Site
  - Home
  - Services
    - System Integration
    - Web Design
    - Web Development
    - Network Administration
  - Products
  - News
  - Partners
  - Community
  - Company
  - Media
  - Examples
  - Mobile
  - Other
  - Special Pages
  - Images

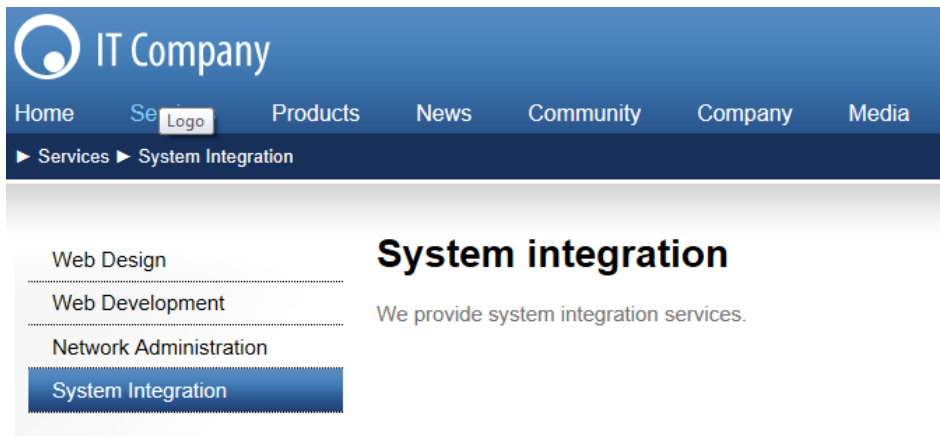
The main editor area shows a preview of the 'System Integration' page. The page has a blue header with the 'IT Company' logo and navigation links: Home, Services, Products, News, Community, Company, Media. The 'Services' menu is expanded to show 'System Integration'. The main content area contains a heading 'System integration' and a paragraph: 'We provide system integration services.'

Now you may want to change the order of the items in the content tree on the left. Click the **Down** button in the main toolbar three times. The *System Integration* item will be moved to the bottom of the section.

- Services
  - Web Design
  - Web Development
  - Network Administration
  - System Integration

Click **Live site** in the main toolbar. You will see your new page as it is displayed to site visitors. Please note that the **System Integration** item is placed at the end of the left menu as you specified.



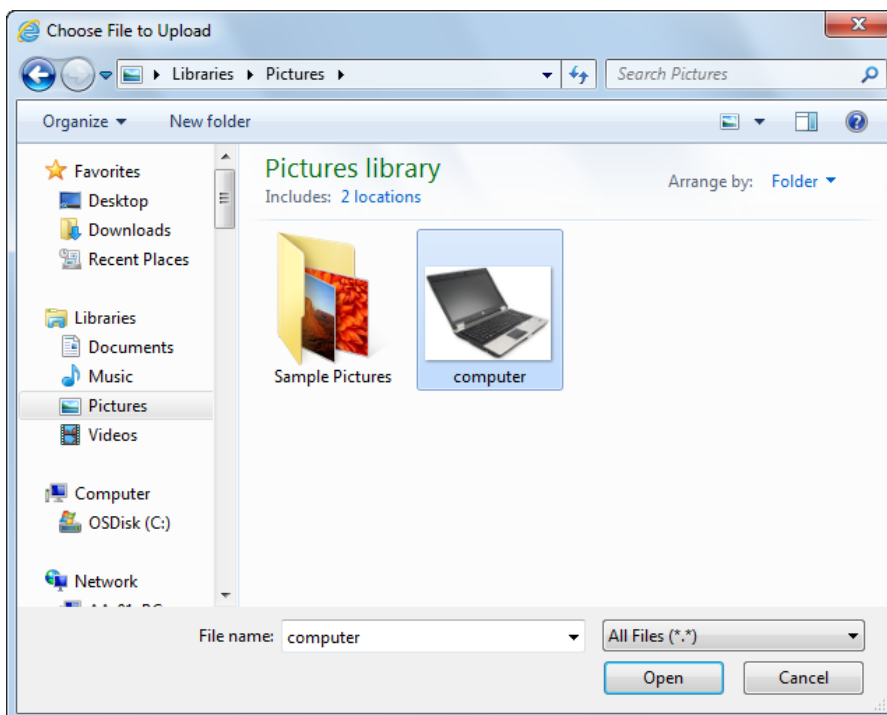


You have learned how to create a new page based on a pre-defined page template.

### 3.4 Inserting an image

Now we will upload and insert a new image to our new page. Click **Services -> System Integration** in the content tree. Switch to the **Edit -> Page** mode. Place the cursor into the main editable region, just below the text, and click the **Quickly insert media** (📎) in the WYSIWYG editor toolbar.

The browser's **Choose file** dialog opens. Locate some suitable image file and click **Open**.



The image will be pasted to the editable region so that the page looks like this:

The screenshot displays the Kentico CMS 7.0 interface. The top navigation bar includes 'Live Site' and 'Site Manager' links. The main toolbar contains 'Save', 'Spell check', and 'Live site' buttons. The content area shows a navigation menu with 'System Integration' selected, and a text block containing the text 'We provide system integration services.' and an image of a laptop.

Click **Save** to save the changes. Click **Live site** to see the new version of your page.

You have learned how to upload an image and insert it into the text.



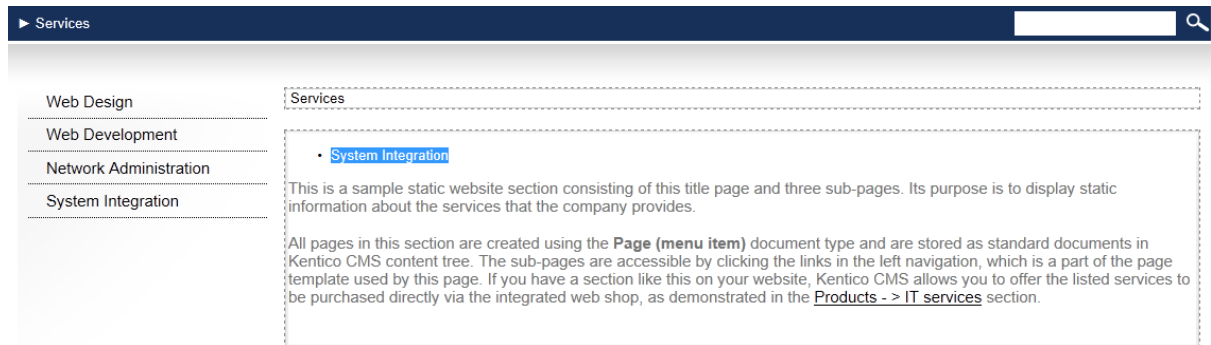
#### Allowing pop-ups for the website


If you are using a pop-up blocker, you may need to allow pop-up windows in your browser so that the Web part properties dialog as well as some other dialogs work correctly. This applies only to the administration interface, so the site visitors are not affected by this.

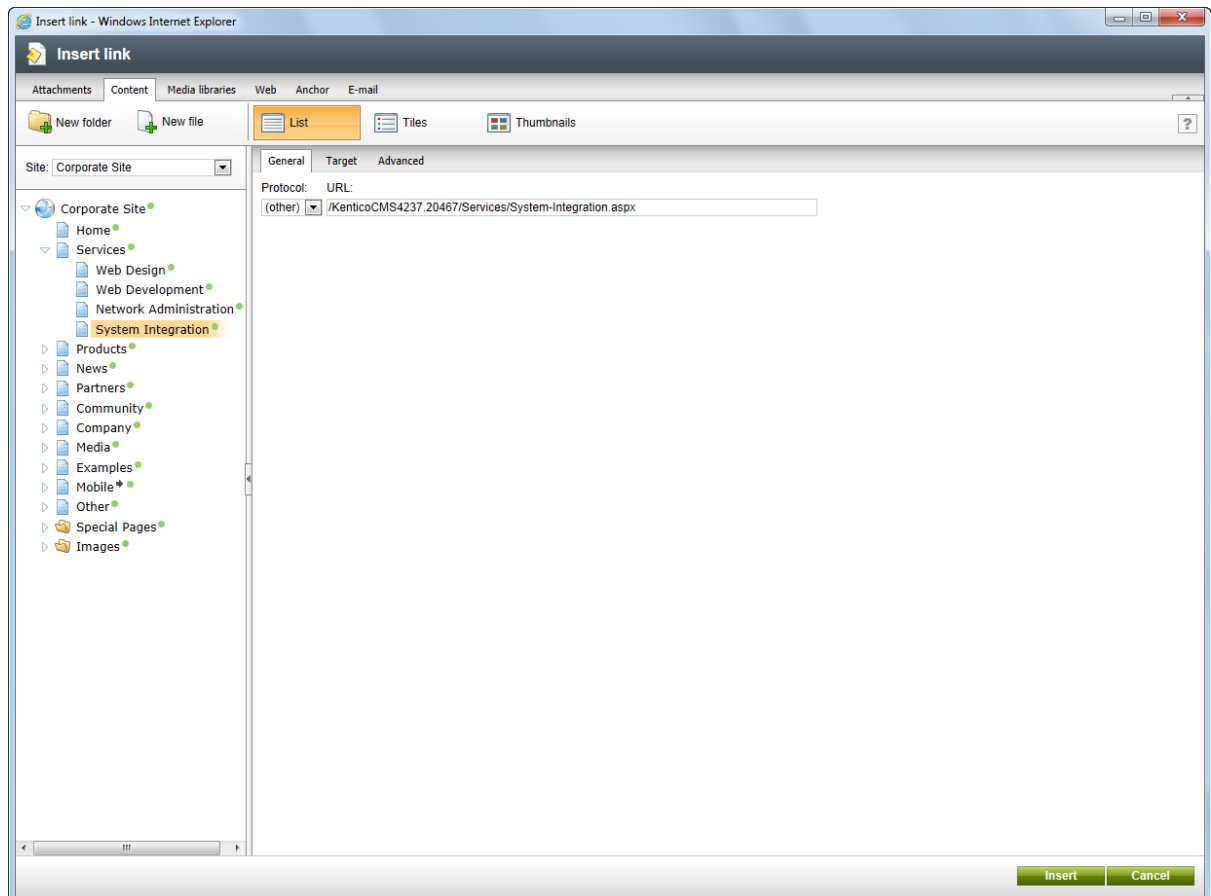
## 3.5 Creating a link

Now we will create a link between the **Services** page and our **System Integration** page. Click **Services** in the content tree and make sure you have the **Edit -> Page** mode selected.


Add a new bulleted list item called **System integration** and select the whole line...



... and click the **Insert/Edit Link** () button in the WYSIWYG editor toolbar. The **Insert link** dialog opens. Switch to the **Content** tab, select the **Services -> System Integration** page from the content tree and click **Insert**.



The text is now marked as a link (underlined).

Click  **Save** and choose the **Live site** mode. Now, when you click on **System integration**, you are redirected to the new page.

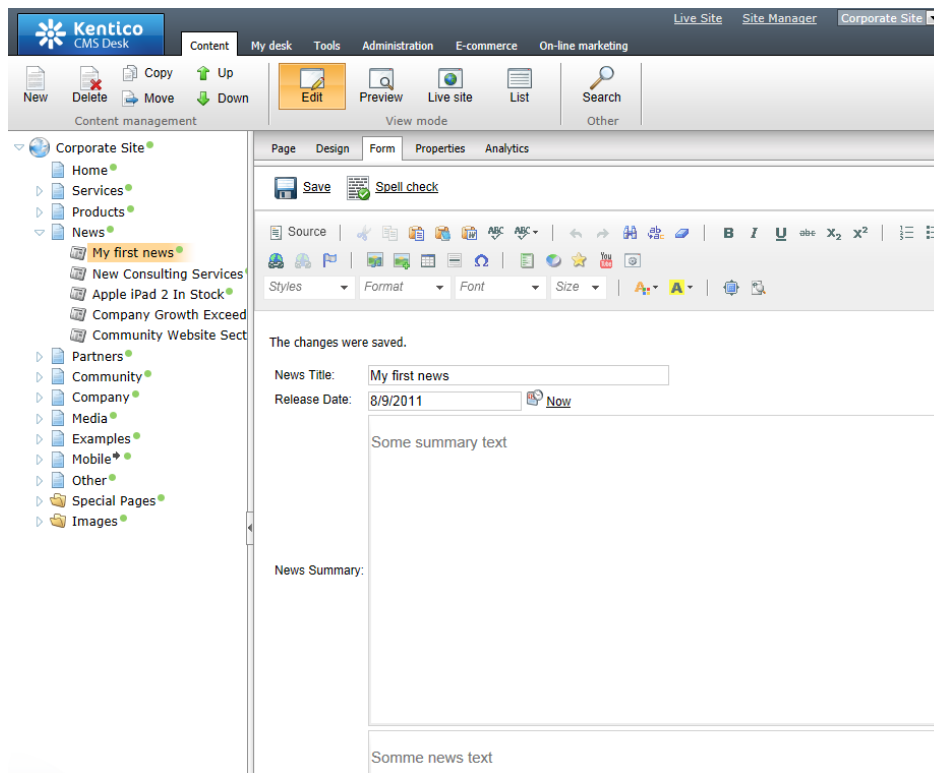
You have learned how to create a link between pages.

### 3.6 Creating a news item

Now you will learn how to create a news item. Click **Edit** in the main toolbar. Click **News** in the content tree and click **New**. Choose to create a new document of type **News**. You are redirected to the form that allows you to define news item sections: title, summary, full text and release date. Enter the following text:

- **News title:** My first news
- **Release date:** click *Now*
- **News summary:** Some summary text.
- **News text:** Some news text.

Click  **Save** to save the new document.



As you can see, the editing mode is now set to **Form** instead of **Page**. It means you do not edit editable regions on the page, but rather the structured data related to the document. The **Form** tab is used for editing the **structured data related to the document**. The document fields are fully customizable for every document type.

When you click **Live site**, you will see the news item displayed using a pre-defined transformation on both the **News** and **News -> My first news** page:



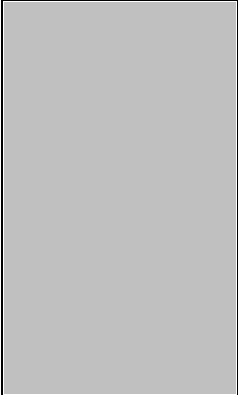
You have learned how to add a news item and how to use the editing form for structured documents.



### Page versus Form

There are two aspects of a document: content stored in editable regions on the page and data stored in form fields. The following table compares both approaches:

	Editable regions on the page	Form
<b>Content structure</b>	Simple content structure, only text content.	Complex content structure, typed data, such as text, date-time, numbers, etc.
<b>Validation</b>	Only basic validation rules for minimum and maximum length.	Complex validation rules, including regular expressions and custom form controls with custom validation code.
<b>Display</b>	The content is displayed in the context of the page as it is displayed in the editing mode.	The content is displayed using XSLT or ASCX transformations using special controls or web parts.
<b>Storage</b>	The content is stored in a single XML document in the document properties.	The content is stored in a separate database table. Each field has its own column. The data can be easily modified using SQL queries or API.
<b>Examples of use</b>	Home page, contact page. Generally: pages with	News, product specification, event details, job openings, etc.

	<p>simply structured or unstructured text-based content.</p> <p>The editable regions are usually used for documents of type <b>Page (menu item)</b>.</p>	<p>Generally: pages with structured content where you need to separate content from design and keep the content in its original data type.</p> <p>Form-based content is usually used for documents of type <b>News, Product, Article</b>, etc.</p>
---	--	--

# Part

---

IV

**Site development overview**

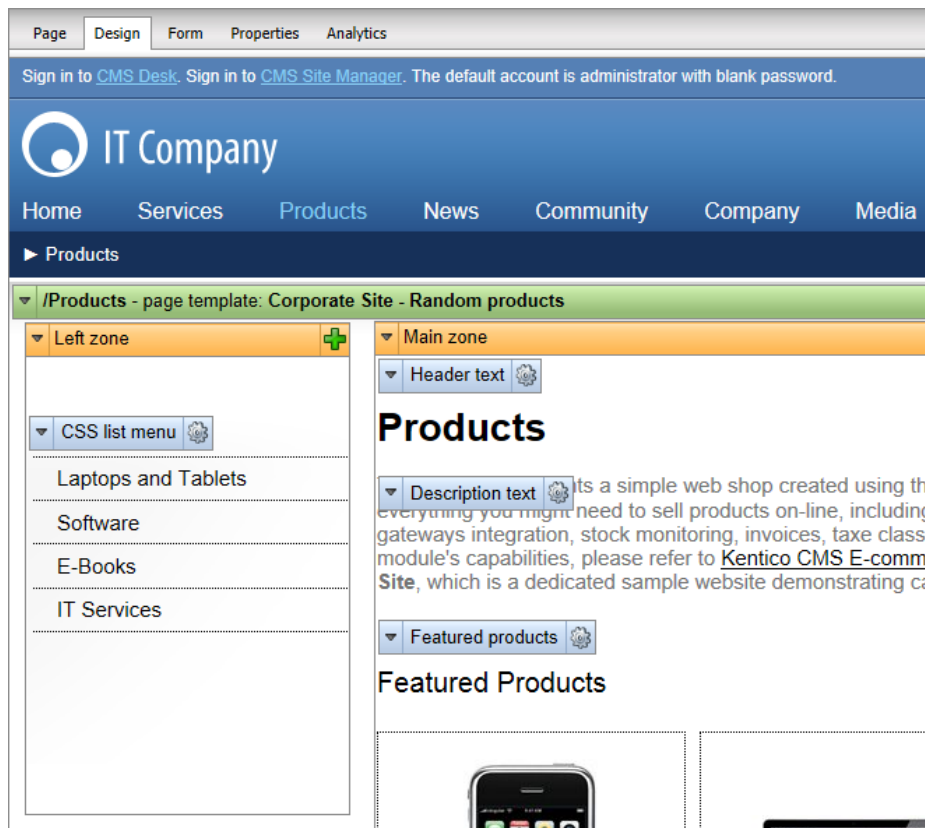
---

## 4 Site development overview

### 4.1 Site development overview

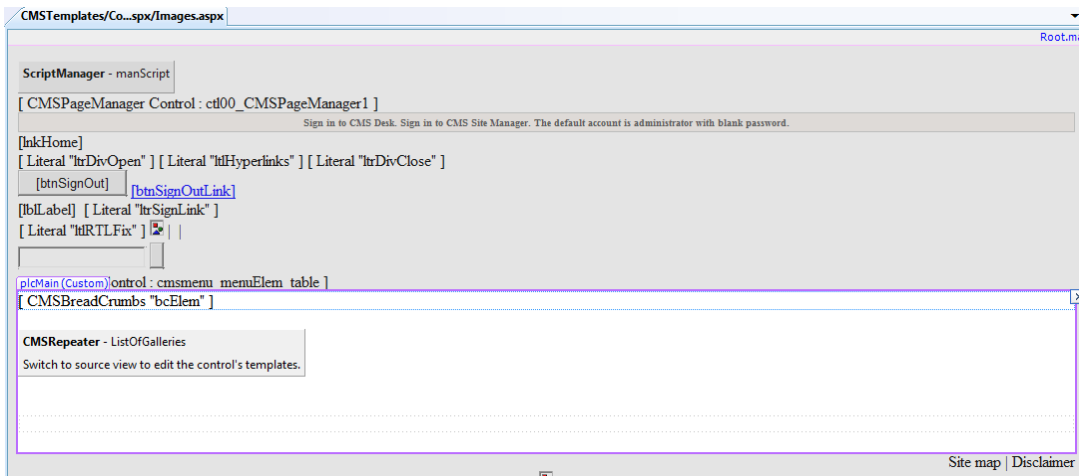
Kentico CMS provides two basic development models and you can choose which one suits you better:

- **Portal Engine** - this model allows you to build websites using a portal engine. It's the recommended way for most developers since it doesn't require programming and using Visual Studio. You can easily build websites using web parts in the **browser-based** user interface.



- **ASPX Templates** - this model can be chosen by advanced ASP.NET developers who prefer to create the website using standard ASP.NET architecture and using standard development tools, such as **Visual Studio**. You need to be familiar with ASP.NET development and have at least basic programming knowledge of C# or VB.NET.





Both approaches are fully supported and they provide the same level of flexibility and extensibility. We recommend using the portal engine, but if you are an advanced .NET developer or wish to integrate existing functionality built on standard ASP.NET architecture, you may want to use ASPX templates. It is also possible to create websites or specific pages using the Model-View-Controller architectural pattern (based on the ASP.NET MVC framework), but this is beyond the scope of this basic tutorial. Please see the [MVC development model](#) chapter in the Developer's Guide for more information.

If required, the available development models can be combined on a single website. For example, you can integrate ASPX templates into a portal engine website, or even custom ASPX pages implementing your own applications. On the other hand, special areas may be defined on ASPX templates where editing can be done through the portal engine.

The following table compares the portal engine and ASPX templates:

	Portal Engine	ASPX Template
<b>How you work</b>	You build websites and design their pages using a browser-based interface.  No programming knowledge is required for common tasks.	You build ASPX pages (web forms) that are used to display content from Kentico CMS.  At least basic programming knowledge of ASP.NET and either C# or VB.NET is required.
<b>How you assemble pages</b>	You use built-in or custom web parts that you place into customizable page layouts.	You use built-in or custom ASP.NET server controls and place them onto the ASPX pages. These are standard ASPX pages that are part of the web project, so you can also work with their code behind files.  It is also possible to place web parts (which are actually ASCX user controls) on the page templates if the required server control is not available.
<b>Master pages and visual inheritance</b>	Sub-pages inherit content from their parent pages by default (so called "visual inheritance"). The inheritance	Page templates may inherit content from a master page, which works just like a standard ASP.NET master page

	can optionally be broken if you want to create a page without parent content.	(.master file).  Pages do not inherit content from their parents in the website content hierarchy, they only inherit from the master page (if it is used).
<b>Custom code integration and extensibility</b>	You can create your own user controls or web parts (ASCX files with a portal engine interface) if you need to integrate custom functionality.  Any custom controls or code can be added to the web parts placed on the website.  You can also use standard ASPX pages within your portal engine-based website.	You build standard ASPX pages with code behind files, which means you can place any custom controls and code onto the page.
<b>Advantages</b>	<ul style="list-style-type: none"> <li>• Easier and faster to build a website.</li> <li>• ASP.NET programming knowledge is not required for common tasks.</li> <li>• You can build the whole website very quickly, using only a web browser.</li> </ul>	<ul style="list-style-type: none"> <li>• Standard ASP.NET architecture.</li> <li>• You can use your favorite development tools, such as Visual Studio.</li> </ul>
<b>Disadvantages</b>	<ul style="list-style-type: none"> <li>• Proprietary architecture and development process.</li> </ul>	<ul style="list-style-type: none"> <li>• Requires ASP.NET programming knowledge.</li> <li>• The design of the web pages cannot be fully managed via the browser-based administration interface.</li> </ul>



### Is Kentico CMS just another portal engine?

Now you may ask what's the difference between Kentico CMS and DotNetNuke or SharePoint.

Well, the main difference is the **flexibility**. Kentico CMS gives you full control over:

- site structure
- site navigation
- page layout
- design
- content structure

Also, it's important to explain that Kentico CMS is a **content management system**, not only a portal engine. It provides features of advanced CMS systems, such as:

- content repository with a logical tree hierarchy of documents
- content/design separation

- custom document types with custom fields
- workflow and versioning
- content locking (check-out, check-in)
- multilingual content
- content preview and content staging
- document-level permissions with permission inheritance
- full-text search in all content
- document management features for uploaded files

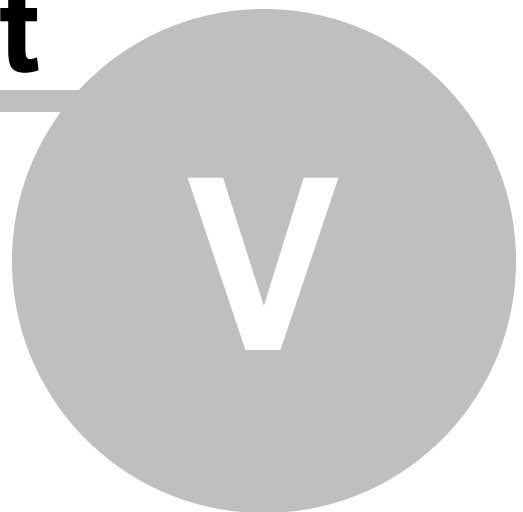
Moreover, Kentico CMS comes with many **professional and flexible built-in modules out-of-the-box**, including Newsletters, On-line forms, Forums, E-commerce, Staging, Image gallery, Event calendar, Events, Blogs, Polls and others

It means you do not need to purchase third-party modules with inconsistent user and programming interface, but you get everything from a single source, with complete documentation.

The rest of this tutorial explains the ASPX template approach. If you want to use the portal engine, please read the Tutorial for the portal engine.

# Part

---



**Creating pages using ASPX templates**

---

## 5 Creating pages using ASPX templates

### 5.1 ASPX page templates

If you are familiar with ASP.NET development in Visual Studio, you may choose to develop websites using standard ASPX page templates. ASPX page templates in Kentico CMS are standard ASP.NET pages that display content from Kentico CMS. They receive the **aliasPath** URL parameter that tells the page template which page should be displayed.

#### What is a page template?

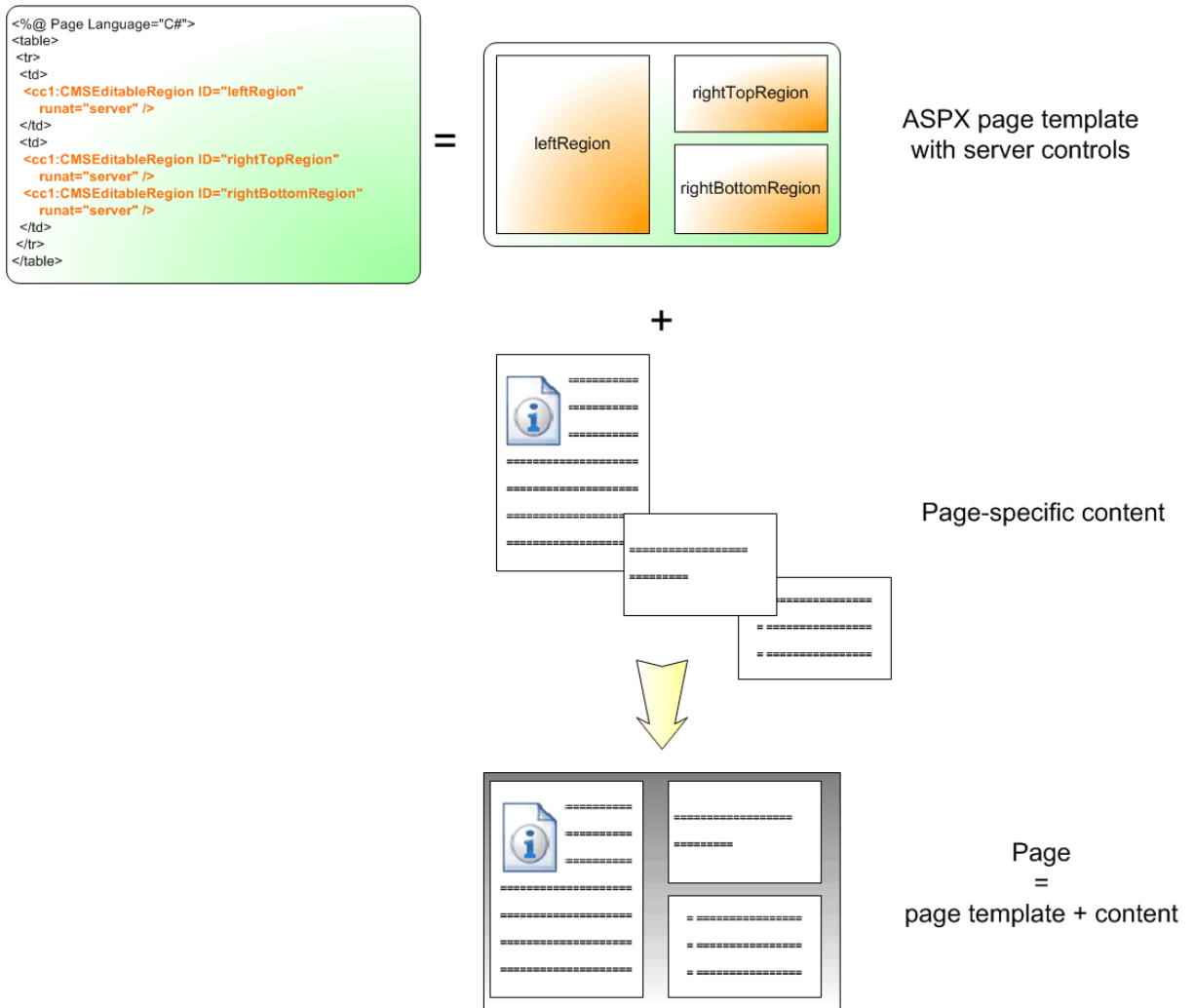
Every web page is based on some page template. The page template can be specific for a single web page ("ad hoc" page template) or it can be re-used for several pages. The following picture shows an example of two pages that use the same page template:



As you can see both of them use the same header, main menu, sub-menu, content structure and footer - they are based on the same **page template**. In this way, you can create multiple pages using the same design.

#### What does the ASPX page template consist of?

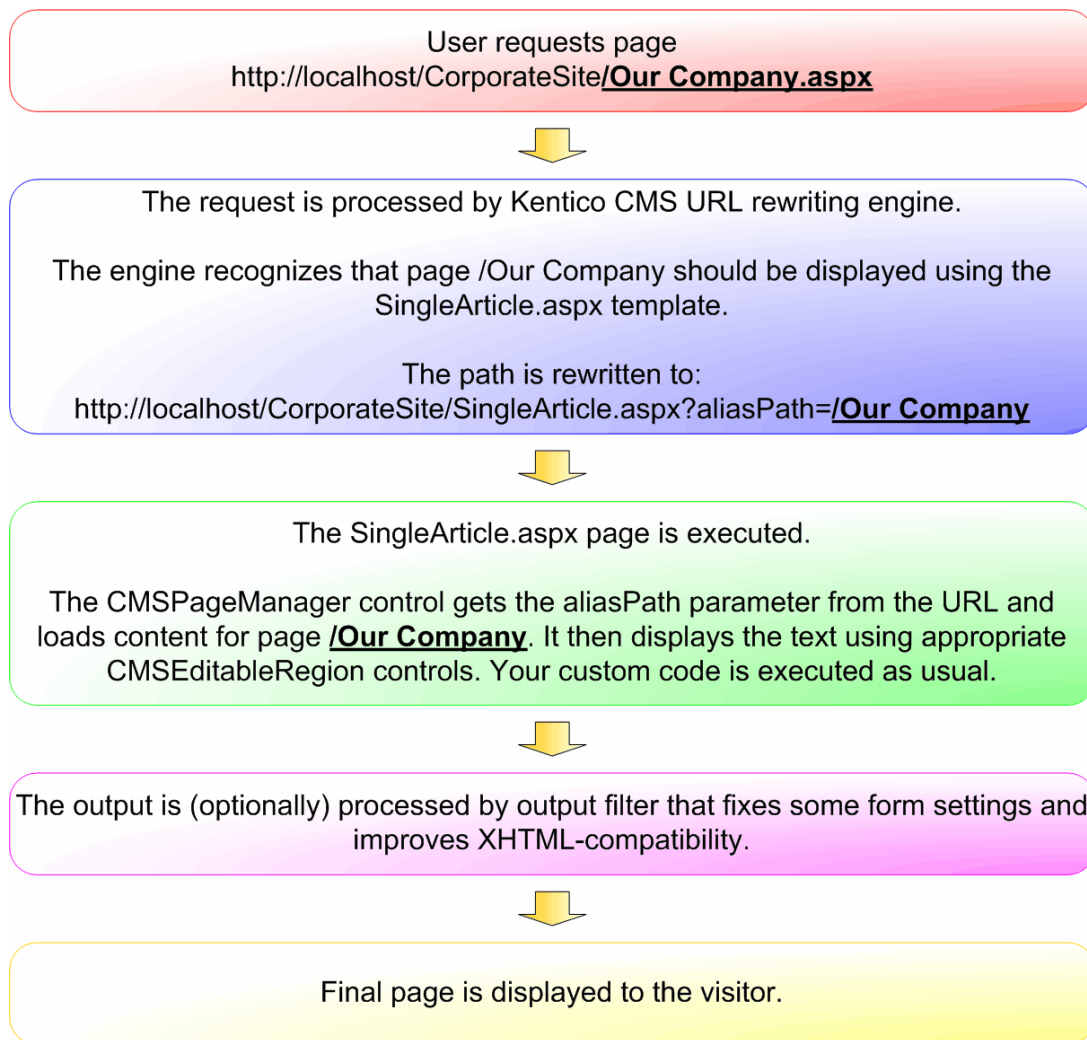
The page template is a combination of static HTML code and ASP.NET server controls (or user controls) that render dynamic content. The following figure illustrates how an ASPX page template and page content are combined to display a page:



As you can see, the ASPX page template is a standard page that may contain HTML code, CMS server controls or any other controls. You can also use code behind (in both VB.NET and C#) to modify page behavior and add custom functionality.

### How is the ASPX page template processed?

When a user requests some page, such as `/services/web-development.aspx`, the system calls the assigned page template with the **aliasPath** URL parameter, which specifies what content (which page from the content tree) should be displayed using the given template:



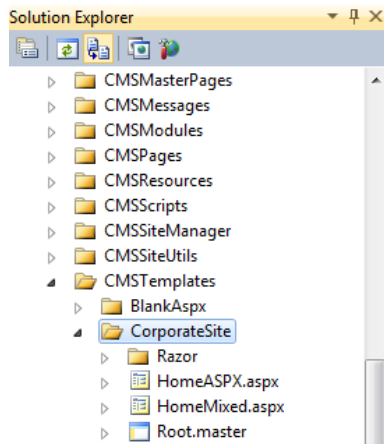
The built-in Kentico CMS controls understand the **aliasPath** parameter in the URL and render the appropriate content automatically.

As you can see, the system uses standard ASP.NET architecture. If you developed the website without Kentico CMS, you would most likely use URLs like this: **/news.aspx?newsid=127** which is similar to **/news.aspx?aliaspath=/news/november news.aspx** as used in Kentico CMS. Kentico CMS also uses friendly URLs in format **/news/your-first-news.aspx** that are better for search engine optimization.

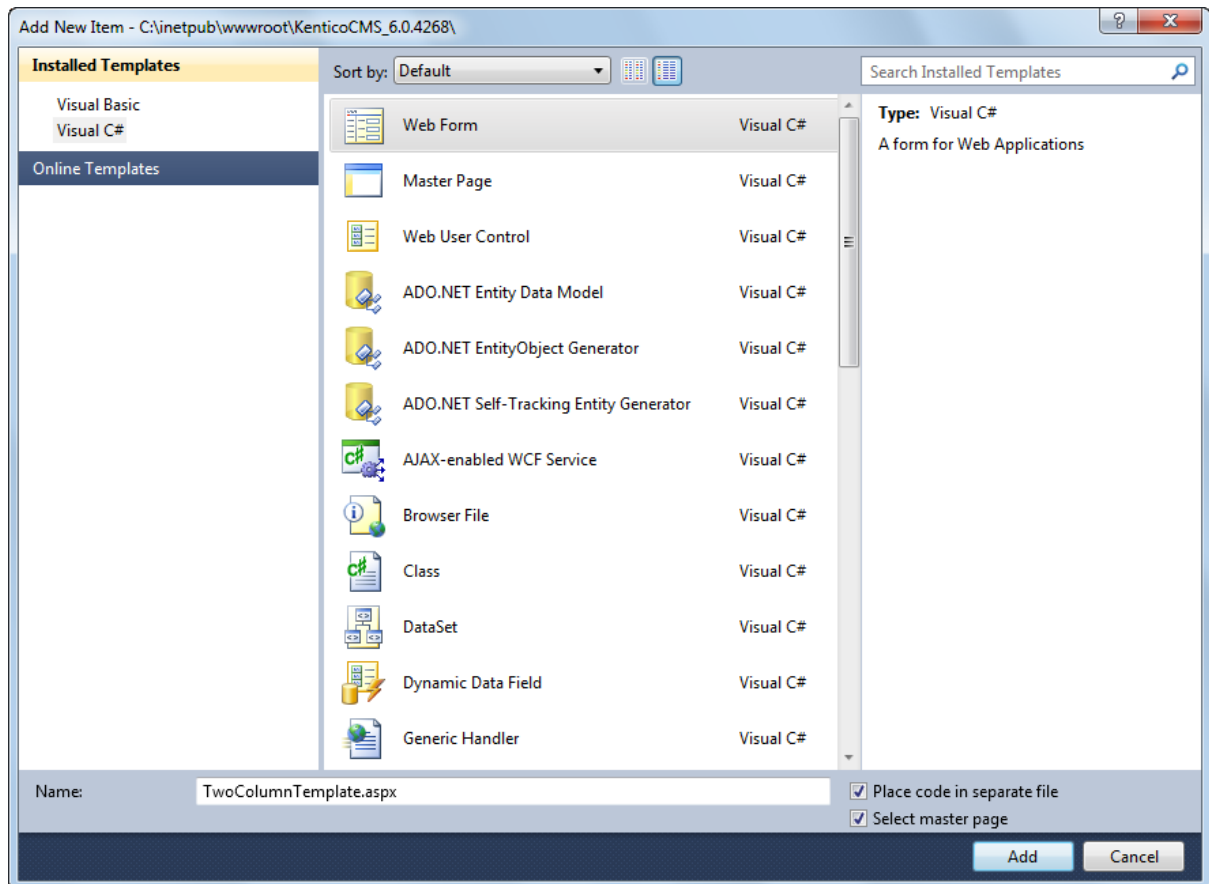
## 5.2 Creating a simple ASPX page template

This topic describes how to create a new ASPX page template. We will create a new page with two columns that contain editable regions.

1. Open the web project in Visual Studio. You can open it either through the **WebProject.sln** file or using the **File -> Open Web Site** option in the menu.
2. Right-click the **CMSTemplates/CorporateSite** folder in the Solution Explorer and select **Add New Item**.



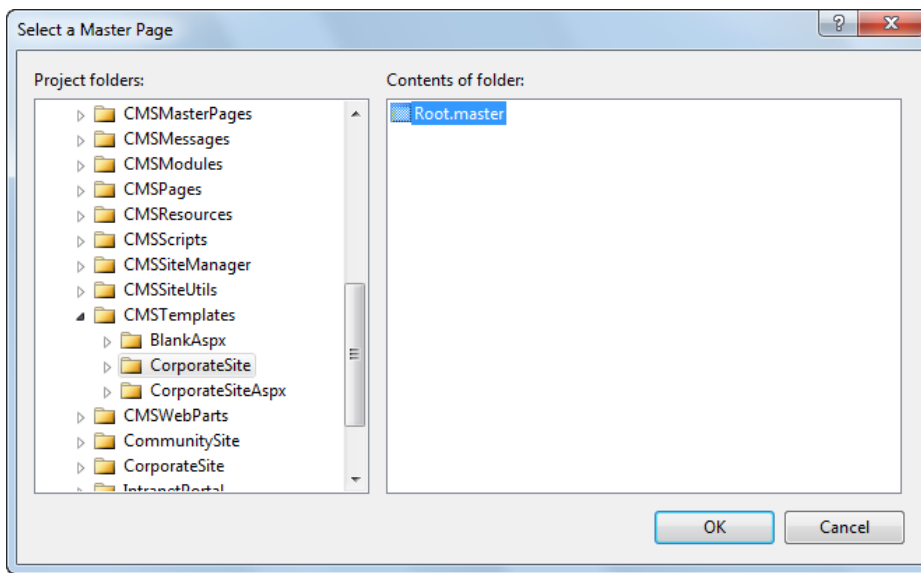
3. Create a new **Web form** named *TwoColumnTemplate.aspx*, check the **Select master page** box and click **Add**.



The **Select a Master Page** dialog opens.

4. Choose the **CMSTemplates/CorporateSite** folder, select the **Root.master** file and click **OK**.





## Writing the ASPX code

5. Open the **Source** view of the newly created ASPX page and add the following code inside the **<asp:Content>** control:

```
<table width="100%">
  <tr>
    <td width="50%">
      <cms:CMSEditableRegion ID="txtLeft" runat="server" DialogHeight="400"
      RegionType="HtmlEditor" RegionTitle="Left column" />
    </td>
    <td width="50%">
      <cms:CMSEditableRegion ID="txtText" runat="server" DialogHeight="400"
      RegionType="HtmlEditor" RegionTitle="Right column" />
    </td>
  </tr>
</table>
```

The **<asp:Content>** control allows you to use the standard ASP.NET concept of master pages. When the system renders the page, it loads the content of the control into the assigned master page (as defined in the *Root.master* file).

The **CMSEditableRegion** control defines an editable region that the page displays as an HTML editor on the **Content -> Edit -> Page** tab of **CMS Desk**. On the live site, the control renders the content entered into the editor.



### Note

This example uses a table layout. If you prefer a CSS layout, replace the surrounding HTML code with <DIV> elements. You have full control over the content.

6. Switch to the code behind file (*TwoColumnTemplate.aspx.cs*) and add a reference to the following namespace:

[C#]

```
using CMS.UIControls;
```

7. Modify the class definition so that it inherits from the **TemplatePage** class:

[C#]



```
public partial class CMSTemplates_CorporateSite_TwoColumnTemplate : TemplatePage
```

Inheriting from this class allows you to use the web form as a page template in Kentico CMS.

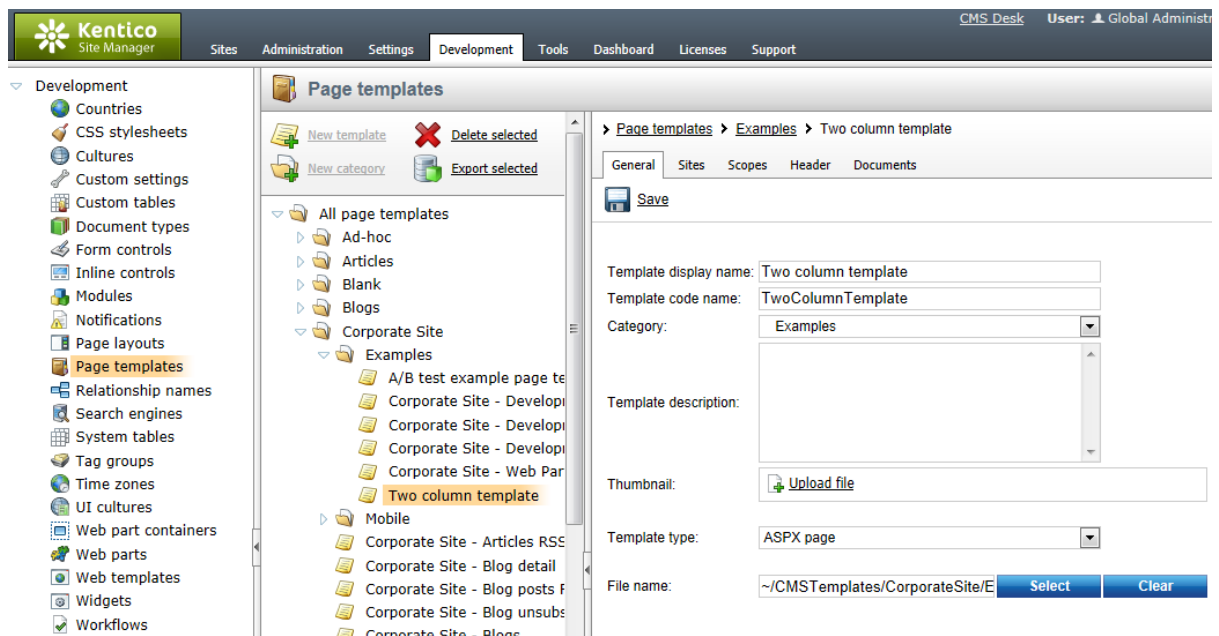
Keep in mind that the name of the class must be identical to the value of the **Inherits** attribute of the `<% @ Page %>` directive on the ASPX page. This is case sensitive.

## Registering the ASPX page as a page template

Now that we have created a new ASPX page, we need to register it in Kentico CMS as a page template, so that it can be used by content editors.

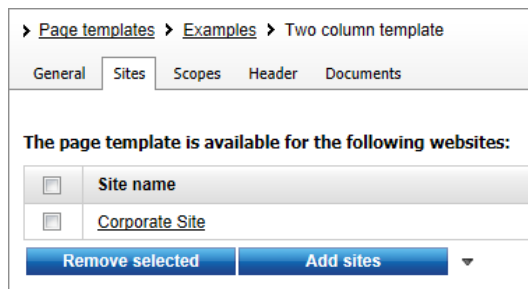
1. Sign in to **Site Manager** and go to **Development -> Page templates**.
2. Select the **Corporate Site/Examples** folder, click  **New template** and type *Two column template* into the **Template display name** field.
3. Click  **Save**. The system creates the template and displays its **General** tab.
4. Set the following values on the **General** tab:
  - **Template type**: ASPX page
  - **File name**: `~/CMSTemplates/CorporateSite/TwoColumnTemplate.aspx`

This is the virtual path of the ASPX page. Alternatively, you can click the **Select** button and manually select the file.



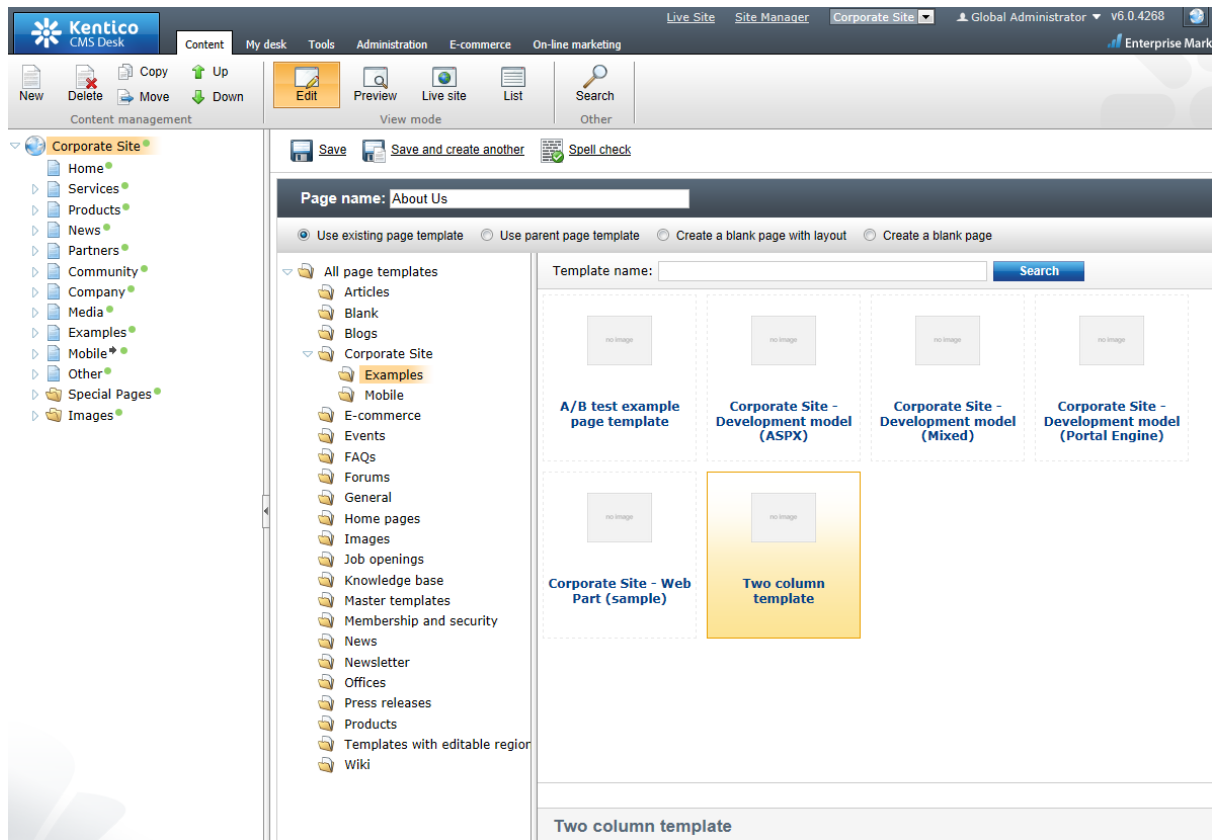
5. Click **Save**.

6. Switch to the **Sites** tab and click the **Add sites** button. Choose the sites where you wish to use the page template and click **OK**.



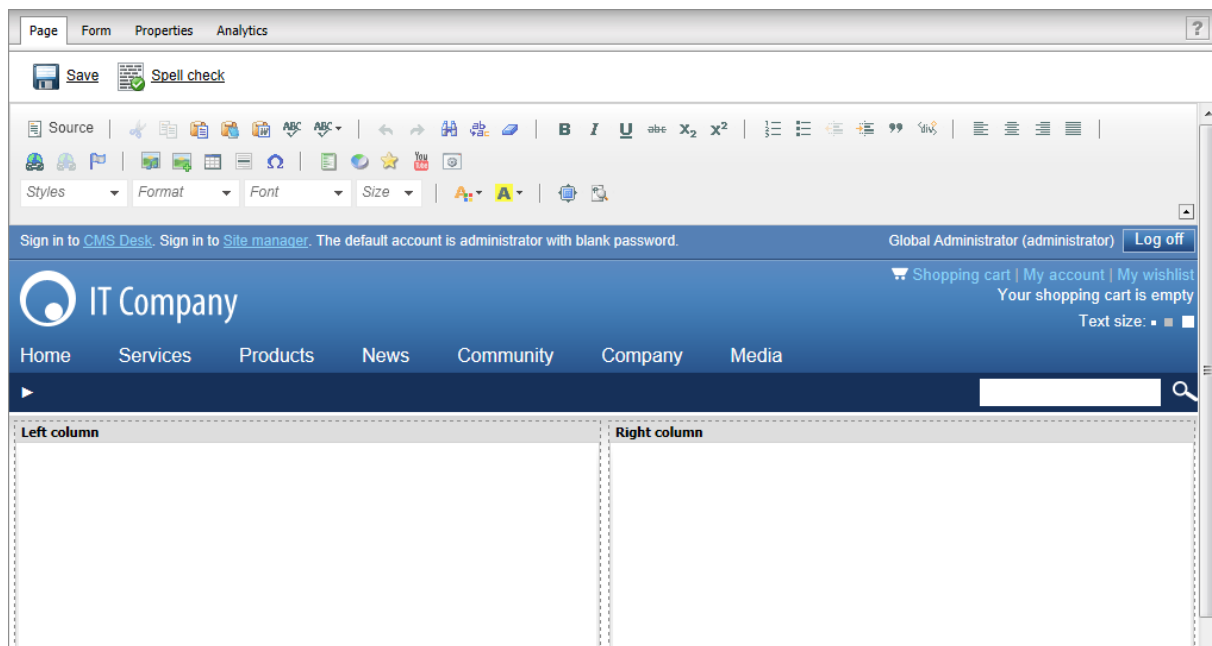
## Creating an About Us page based on the new page template

1. Go to Kentico **CMS Desk** -> **Content**.
2. Select **Corporate Site** (the root of the content tree) and click **New** in the main menu of the **Content** section.
3. Choose the **Page (menu item)** document type.
4. Type *About Us* into the **Page name** field and choose the **Use existing page template** option. Select the **Corporate Site/Examples** category and the **Two column template** page template.




5. Click **Save** to create the new page.

On the **Page** tab, you can see the page and its editable regions.



You have just created your first page based on an ASPX page template. Now you can enter some text

and click  **Save** to store the content.



### Moving pages

If you want to move the About Us page to another position in the menu, you can either use the **Up** and **Down** arrows on the main toolbar or drag the page to the desired location in the content tree.

## 5.3 Using master pages

You can use standard ASP.NET [master pages](#) together with ASPX page templates. This is a powerful concept that allows you to share content across all pages without having to add it separately to every page template. For example, you can create master pages containing header and footer sections with a logo, navigation menu, search box etc.

Master pages are defined in files with the **.master** extension. You can assign one master page to every ASPX page. Master pages must always contain one or more **ContentPlaceHolder** controls as shown here:

```
<asp:ContentPlaceHolder ID="plcMain" runat="server"></asp:ContentPlaceHolder>
```

The **ContentPlaceHolder** control specifies where child pages display their content inside the master page.

It is recommended to store master pages in the **CMSTemplates** folder together with the page template files, so that the system can export them along with your website when you deploy it to another instance of Kentico CMS.

### Creating master pages for ASPX templates

The following code sample shows the markup of a basic master page.



### Important!

If you installed the Kentico CMS project as a web application, you need to rename the *CodeFile* attribute on the first line to *Codebehind* for the code example to be functional. This attribute's value needs to match the name of the master page's code behind file.

Set the value of the *Inherits* attribute according to the location and name of the master page file.

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="Custom.master.cs"
```

```

Inherits="CMSTemplates_CorporateSite_Custom" %>
<%=DocType%>
<html xmlns="http://www.w3.org/1999/xhtml" <%=XmlNamespace%>>
<head id="Head1" runat="server">
  <title id="Title1" runat="server">My site</title>
  <asp:literal runat="server" id="ltrlTags" enableviewstate="false" />
</head>
<body class="<%=BodyClass%>" <%=BodyParameters%>>
  <form id="form1" runat="server">
    <asp:Placeholder runat="server" ID="plcManagers">
      <ajaxToolkit:ToolkitScriptManager ID="manScript" runat="server"
EnableViewState="false" ScriptMode="Release" />
      <cms:CMSPortalManager ID="CMSPortalManager1" runat="server"
EnableViewState="false" />
    </asp:Placeholder>

    <cms:CMSMenu ID="cmsmenu1" runat="server" Cursor="Pointer"
HighlightAllItemsInPath="true" Layout="Horizontal" Padding="0" Spacing="1" />

    <asp:ContentPlaceholder ID="plcMain" runat="server">
    </asp:ContentPlaceholder>
  </form>
</body>
</html>

```

All ASPX page templates require the following manager controls, so it is a good practice to add them onto your website's master page:

- **ajaxToolkit:ToolkitScriptManager** - allows pages to use AJAX components. This is required by all pages that contain the *CMSPortalManager* control.
- **CMSPortalManager** - ensures the transferring of content between the database and editable regions. It also provides the management functionality needed for portal engine zones. Always place this control after the *ToolkitScriptManager* control.



#### **CMSPageManager control**

You may use the **CMSPageManager** control instead of the *CMSPortalManager*. This control is an older equivalent available for the purposes of backwards compatibility. It does not support ASPX templates containing portal engine web part or widget zones.

The **CMSMenu** control is one of the options that you can use to generate a drop-down menu for website navigation.

### **Writing the master page code behind**

You also need to modify the code behind file of the master page:

1. Add a reference to the **CMS.UIControls** namespace:

**[C#]**

```
using CMS.UIControls;
```

**[VB.NET]**

```
Imports CMS.UIControls
```

2. Change the class definition to match the following (the name of the class may be different):

**[C#]**

```
public partial class CMSTemplates_CorporateSite_Custom : TemplateMasterPage
```

**[VB.NET]**

```
Partial Class CMSTemplates_CorporateSite_Custom  
    Inherits TemplateMasterPage
```

Master pages must always inherit from the **TemplateMasterPage** class.

3. Add the following code into the master page's code behind class:

**[C#]**

```
protected override void CreateChildControls()  
{  
    base.CreateChildControls();  
    PageManager = CMSPortalManager1;  
}  
  
protected override void OnPreRender(EventArgs e)  
{  
    base.OnPreRender(e);  
    this.ltlTags.Text = this.HeaderTags;  
}
```

**[VB.NET]**

```
Protected Overrides Sub CreateChildControls()  
    MyBase.CreateChildControls()  
    PageManager = CMSPortalManager1  
End Sub
```

```
Protected Overrides Sub OnPreRender(e As EventArgs)
    MyBase.OnPreRender(e)
    Me.ltlTags.Text = Me.HeaderTags
End Sub
```

Adjust the value of the **PageManager** property according to the ID of the *CMSPortalManager* (or *CMSPageManager*) control placed on the master page.

This code ensures that ASPX templates using the given master page support all required functionality.

## 5.4 Adding portal engine functionality to ASPX templates

When developing or maintaining a website using ASPX page templates, one of the drawbacks is that the code of a page must be modified manually every time you wish to change its design. It is possible to add flexibility to ASPX templates by defining areas that may be edited directly through the browser in the Kentico CMS administration interface. The techniques used to manage these areas are the same as those used to design portal engine page templates. To learn more about portal engine features, please read the version of this tutorial dedicated to the portal engine.

### Example

In this example, you will learn how to create a simple ASPX page template with areas that can be designed via the portal engine.

### Writing the ASPX code

1. Open the web project in Visual Studio again (using either the **WebProject.sln** file or **File -> Open -> Web site** in the menu).
2. Right-click the **CMSTemplates -> CorporateSite** folder in the Solution Explorer and select **Add new item**.
3. Create a new **Web form** named *TwoZones.aspx*, check the **Select master page** box and click **Add**. In the master page selection dialog, choose the **Root.master** page from the **CMSTemplates/CorporateSite** folder and click **OK**.
4. Open the **Source** view of the newly created ASPX page and place the following code inside the **<asp:Content>** control:

```
<cms:CMSPagePlaceholder ID="plcZones" runat="server">
    <LayoutTemplate>

        <table width="100%" cellpadding="0" cellspacing="0">
            <tr valign="top">
                <td width="50%">
                    <cms:CMSWebPartZone ID="zoneLeft" runat="server" />
                </td>
                <td width="50%">
```



```
<cms:CMSWebPartZone ID="zoneRight" runat="server" />
</td>
</tr>
</table>

</LayoutTemplate>
</cms:CMSPagePlaceholder>
```

The **CMSPagePlaceholder** control creates an area on the page that behaves in a way similar to a portal engine page template.

The **<LayoutTemplate>** element defines the layout of the area. This example uses a basic two column table structure, but setting a CSS-based layout applied through HTML elements (e.g. **<div>**, **<span>**, etc.) is also a valid option.

The table contains two **CMSWebPartZone** controls, which represent fully functional portal engine zones. Users can manage these zones when editing a page based on the page template on the **Edit -> Design** tab of **CMS Desk**. When some content is defined for a zone, the system stores the information in the database along with the respective page template object, not in the actual code of the ASPX page. Communication with the database is ensured by the **CMSPortalManager** control, which is located on the **Root.master** page.

5. Switch to the code behind file (*TwoZones.aspx.cs*) and add a reference to the following namespace:

[C#]

```
using CMS.UIControls;
```

6. Modify the class definition to inherit from the **TemplatePage** class as shown below:

[C#]

```
public partial class CMSTemplates_CorporateSite_TwoZones : TemplatePage
```

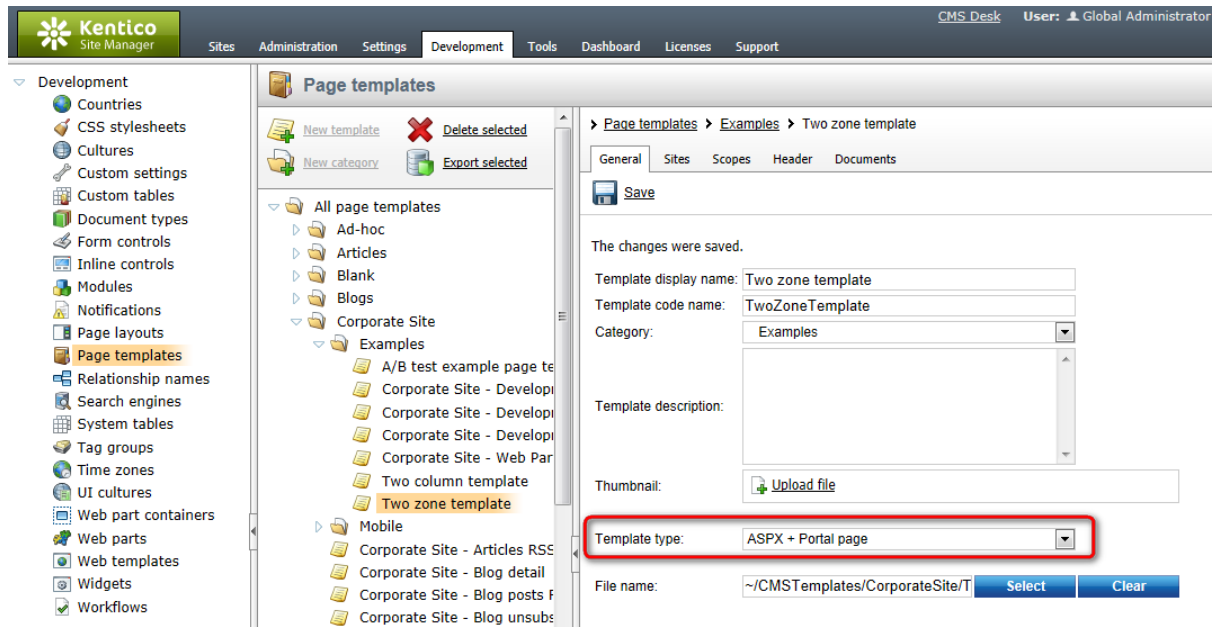
7. Save the changes made to both the ASPX page and its code behind file.

Now you can use the page as a page template in Kentico CMS.


## Registering the ASPX page as a page template

1. Sign in to **Site Manager** and go to **Development -> Page templates**.
2. Select the **Corporate Site/Examples** folder, click  **New template** and type *Two zone template* into the **Template display name** field.
3. Click  **Save**. The system creates the template and displays its **General** tab.
4. Select the **ASPX + Portal page** option for the **Template type** property. This is necessary in order for

the **Design** tab of **CMS Desk** to be available when editing pages using the template.



5. Enter the following path into the **File name** field: `~/CMSTemplates/CorporateSite/TwoZones.aspx`

6.  **Save** the changes.

7. Switch to the **Sites** tab and use the **Add sites** button to assign the page template to the site that you are using (Corporate Site).


## Using ASPX + Portal engine templates

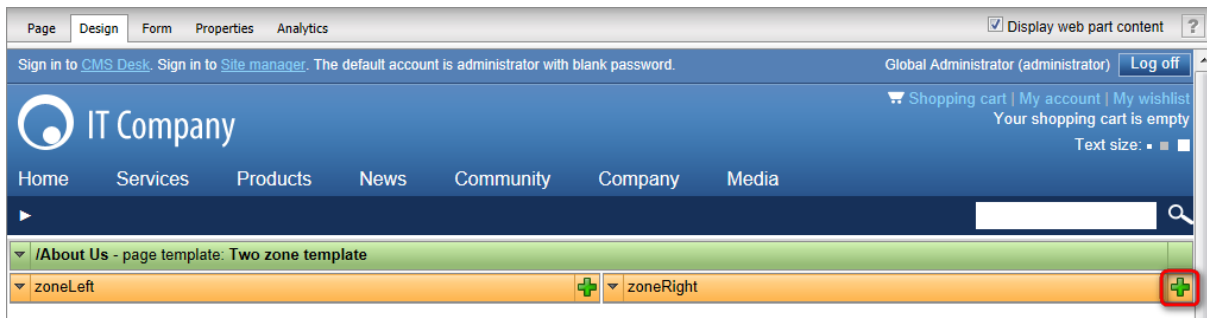
We will modify the **About Us** page created in the [previous example](#) to use the new page template.

1. Go to **CMS Desk -> Content**, select the *About Us* page and switch to its **Properties -> Template** tab.

2. Click the **Select** button and choose the **Corporate Site/Examples/Two zone template** page template from the catalog. Click  **Save** to confirm the page template change.

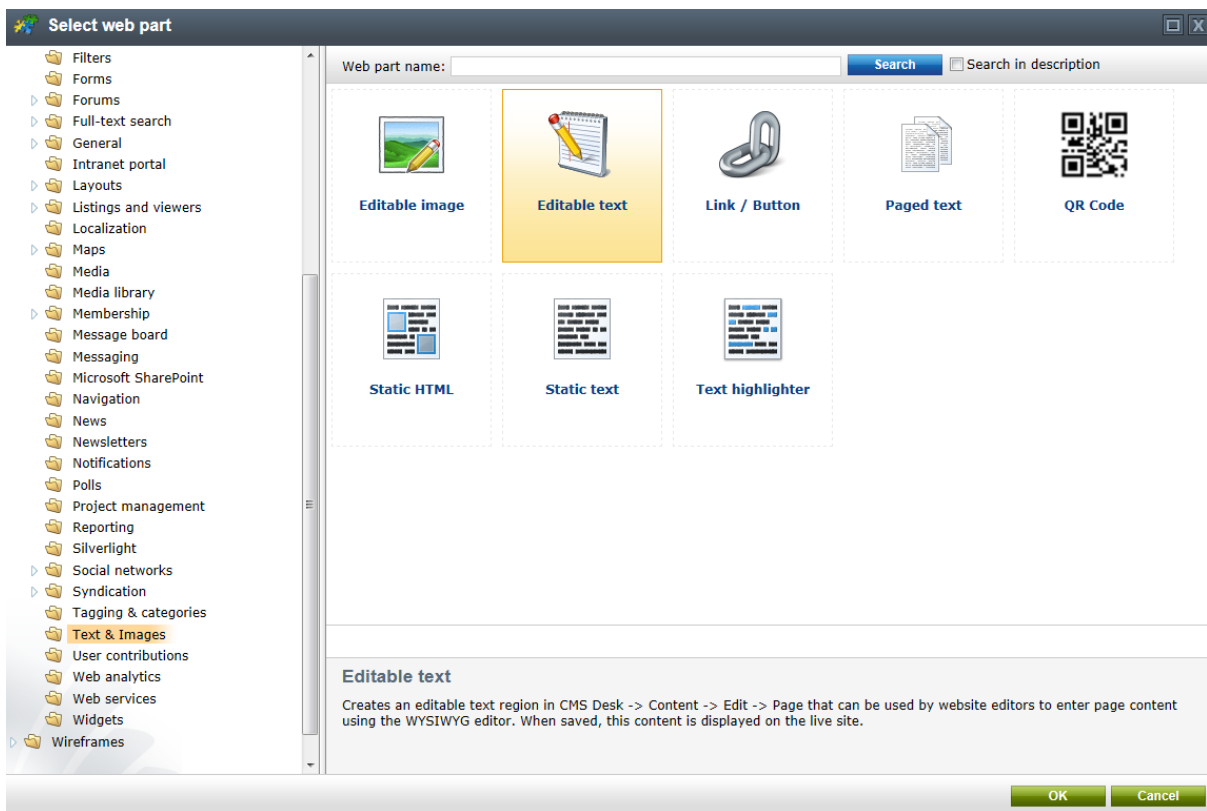
3. Refresh your browser window and switch to the **Design** tab, which is now available for the **About Us** page. You can see two empty zones on the page as defined in the ASPX code of the template. To define the content of standard zones, add components called web parts.

4. Add a web part to the **zoneRight** zone by clicking the **Add web part**  icon in the top right corner.



The **Select web part** dialog opens.

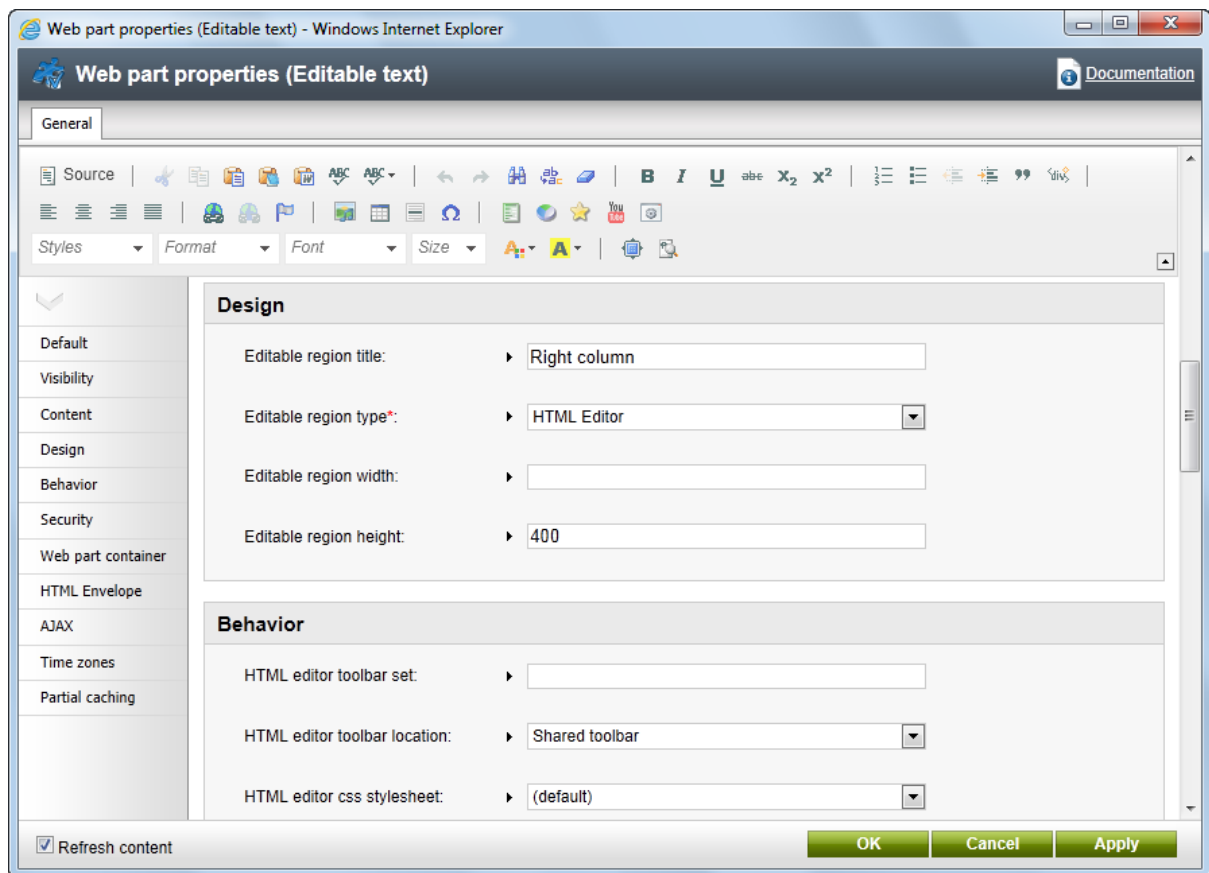
5. Choose the **Text & Images -> Editable text** web part and click **OK**.



The **Web part properties** dialog opens, which allows you to configure the web part.

6. Enter the following values:

- **Design -> Editable region title:** Right column
- **Design -> Editable region height:** 400

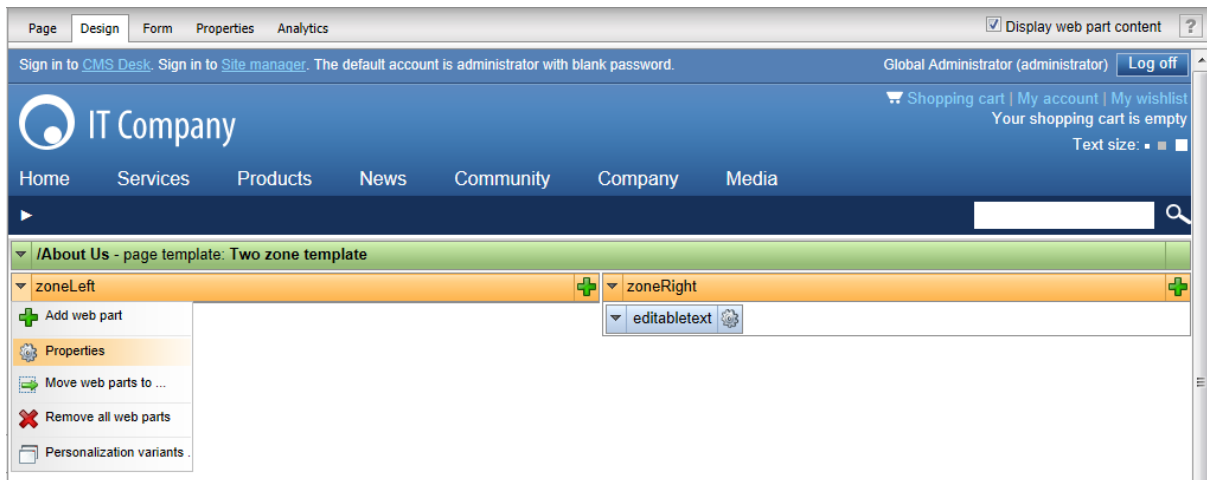


Click **OK** to save the web part and add it to the zone.

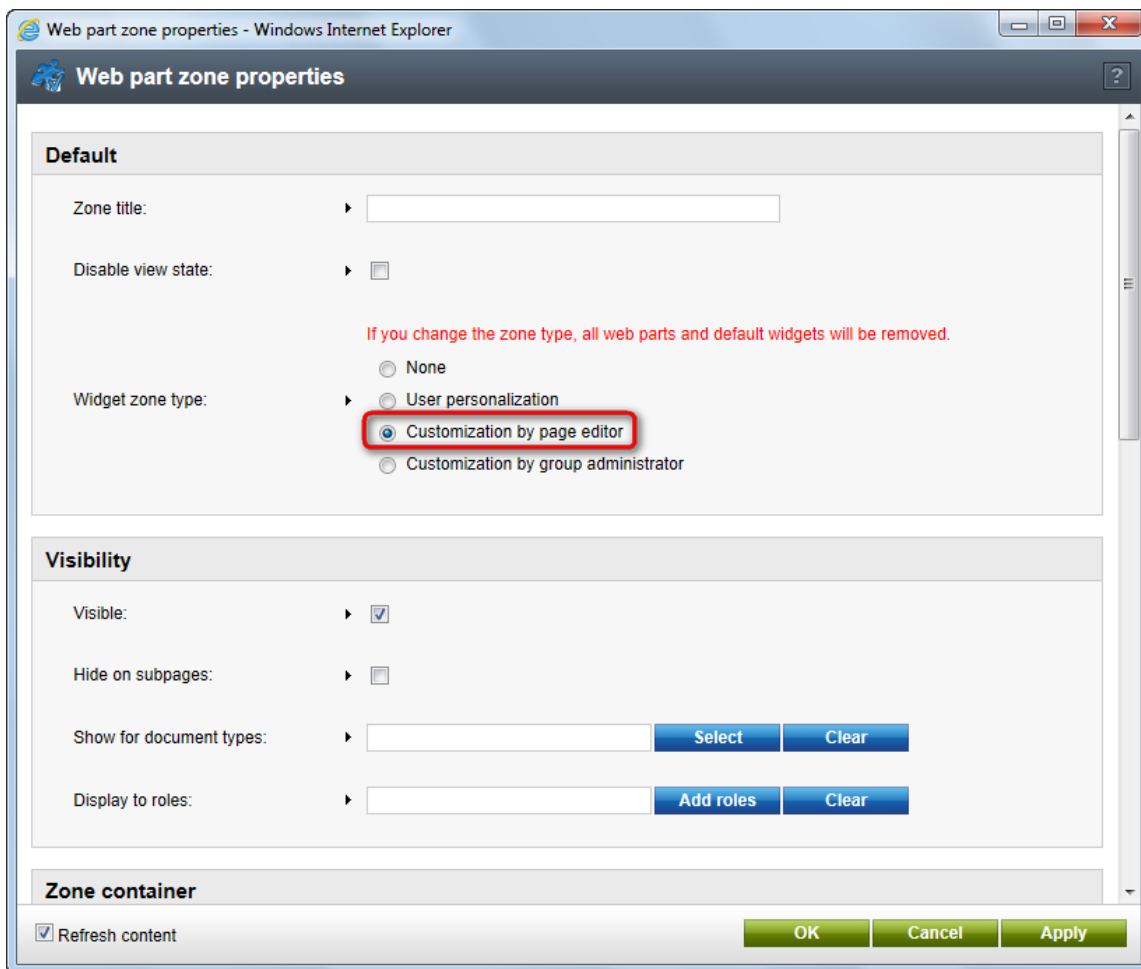
This web part provides a text area on the page that users can edit on the **Page** tab of **CMS Desk**, just like the editable regions in the previous example. As you can see, the template allows you to build the design of the page using a browser-based interface. A web part zone may contain any number of web parts.

You may also configure zones to use various types of widgets, which are objects similar to web parts, but they allow page customization by different kinds of website users, not just the administrators or designers.

7. Expand the menu (▼) of the **zoneLeft** zone and select **Properties** to configure the zone.



8. Switch the **Widget zone type** property from *None* to *Customization by page editor*.



Click **OK**. This zone now serves as a widget zone for page editors.

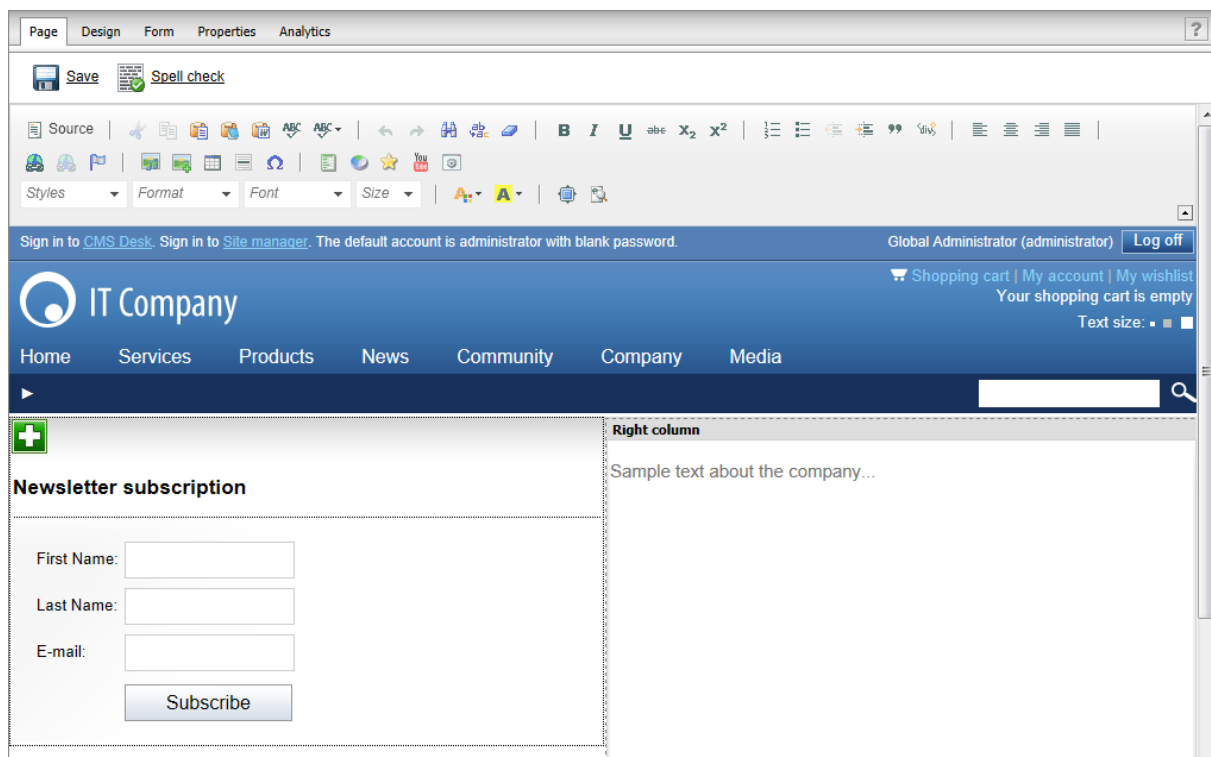
9. Switch to the **Page** tab, enter some content into the editable text region displayed by the web part on the right and click **Save**.

You can also manage the editor widget zone on the left.

10. Click the **Add widget** (+) button in the top left corner of the widget zone to place a widget onto the page. Select the **Newsletters -> Newsletter subscription** widget from the catalog and set the following values for its properties:

- **Newsletter name:** Corporate Newsletter
- **Allow user subscribers:** disabled (unchecked)
- **Widget container:** Corporate site - Light gradient box
- **Widget container title:** Newsletter subscription

Click **OK** to add the widget to the page. It provides a form which users may use to subscribe to the site's newsletter.



The example demonstrates how to use web parts or widgets to build the design of pages based on ASPX page templates. This approach combines the standard architecture and development process of ASPX templates with the flexibility and user-friendliness of the portal engine.

# Part

---



VI

**Managing styles and design**

---

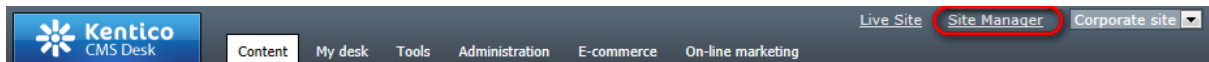
## 6 Managing styles and design

### 6.1 CSS styles

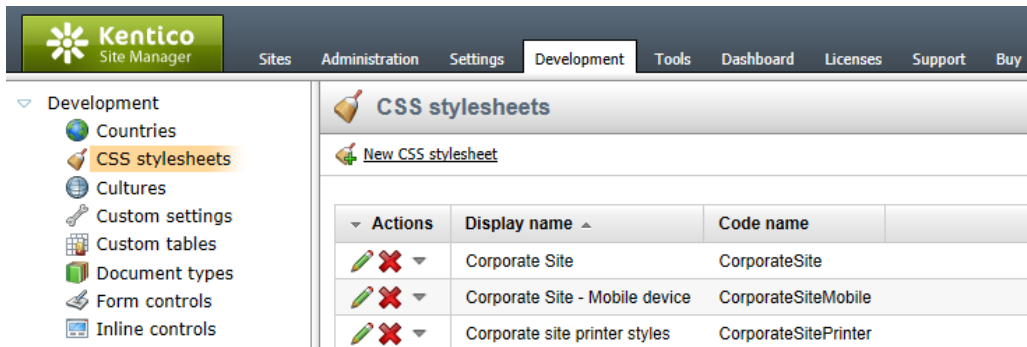
The design of the website relies on standard CSS styles. Each website has its global CSS stylesheet that can be selected in **Site Manager -> Sites -> ... edit site ... -> General**. Here you can also choose a different stylesheet for the site's WYSIWYG editors.

In addition, each page can override the global website stylesheet. You can assign a stylesheet to a specific page by editing its corresponding document in **CMS Desk -> Content -> Properties -> General** and using the **CSS stylesheet** selector. The actual content of the chosen stylesheet can be accessed by clicking the **Edit** button.

All stylesheets in the system may also be managed in Site Manager. When you are in CMS Desk, you can easily switch to Site Manager by clicking the **Site Manager** link in the header:

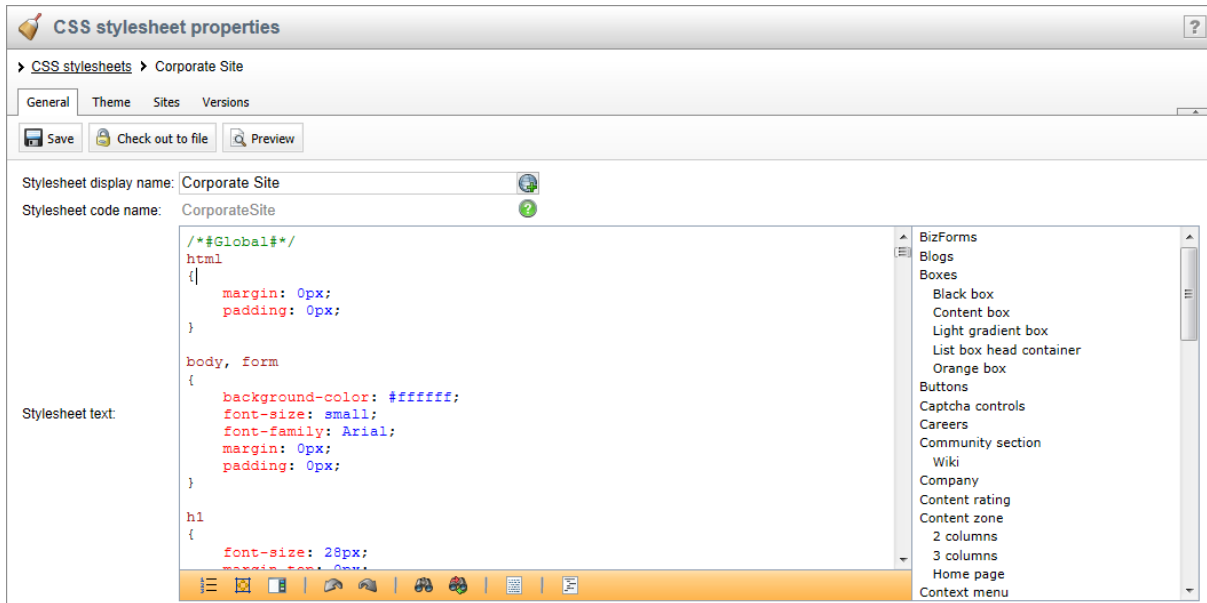



Then click **Development** in the **Site Manager** main menu and select **CSS stylesheets** from the left menu:



**Edit** () the **Corporate Site** stylesheet:





When modifying a stylesheet, you can use the  **Preview** button in the header to view the website's pages side-by-side with the CSS code. This allows you to immediately check how changes affect the appearance of the site. Please note that while the preview works immediately, you may need to clear your browser cache before the new CSS design is reflected on the actual live site.



### Browser and language-specific styles

Pages automatically have CSS classes assigned to their `<body>` element according to the characteristics of their language (its text direction and specific culture) and depending on the browser in which they are currently viewed. For example:

```
<body class="LTR IE IE9 ENUS">
```

As you can see, four types of classes are added:

- **Text direction** - the *LTR* class is assigned for left-to-right languages and *RTL* for right-to-left.
- **Browser type** - this class is added according to the browser in which the page is currently opened. The following classes are used:

Browser	Class name(s)
Internet Explorer	IE
Firefox	Gecko
Google Chrome	Chrome, Safari
Safari	Safari
Opera	Opera

- **Browser version** - the class name is the same as for the browser type, but with the number of the browser's major version appended, e.g. *IE9*, *Gecko5* etc.
- **Culture** - the name of the class is added based on the culture code of the page's content (without the hyphen), for example *ENUS* for pages using the *en-US* culture.

This feature allows you to style page elements differently according to the browsing environment of the current visitor. You can define styles for any combination of the classes mentioned above.

For example, you can add the following into a website's stylesheet:

#### [CSS]

```
.IE8 .MyFont
{
    font-size: 20px;
}

.Opera .MyFont
{
    font-size: 18px;
}
```

Now elements styled using the *MyFont* class will have a different font size when viewed in the Internet Explorer 8 or Opera (all versions) browsers.

## 6.2 App themes

In some cases, you may leverage the built-in support for [ASP.NET themes](#). You can use them to set styles for controls that do not have their own CSS class name, such as Datagrid, Calendar or web parts with complex dialogs (logon form, registration form, ...).

Themes must be defined in a folder located under the **App\_Themes** directory. The name of this folder needs to be the same as the code name of the site's CSS stylesheet. So if you use the *Green* stylesheet on your site, your themes must be stored in the *App\_Themes\Green* sub-folder.

Skins for your controls must be added to the **Default.skin** file under this folder. Here's an example of a skin for the **CMSCalendar** control / **Calendar** web part:

```
<cms:CMSCalendar Runat="server">
    <NextPrevStyle ForeColor="Red"></NextPrevStyle>
    <WeekendDayStyle BackColor="#E0E0E0"></WeekendDayStyle>
</cms:CMSCalendar>
```

The code above defines the appearance of the calendar control. You can see this control on the Events page of the sample Corporate Site.



### Where should I store website design files?

It's recommended to store all images, flash movies and other resources that are part of the website design template in the **App\_Themes/<stylesheet code name>** folder. This ensures that the files are exported together with the stylesheet if you deploy it to some other server.

## 6.3 Menu design

Now you will learn how to change the design of the main navigation menu. The main menu used on the sample Corporate Site is displayed using the **CSS list menu** web part which is based on the **CMSListMenu** server control.

The menu design depends primarily on the applied CSS styles. Here's an example of the CSS styles used for the drop-down menu:

[CSS]

```
.zoneMenu .CMSListMenuUL
{
    list-style: none;
    margin: 0px;
    padding: 0px;
    position: relative;
}



.zoneMenu .CMSListMenuUL li
{
    float: left;
    padding: 0px 22px 0px 0px;
}

.zoneMenu .CMSListMenuUL li a
{
    color: #fff;
    text-decoration: none;
    display: block;
    height: 23px;
    font-size: 16px;
    line-height: 23px;
    padding: 0px 8px;
    border: 1px solid transparent;
    font-family: Arial;
}

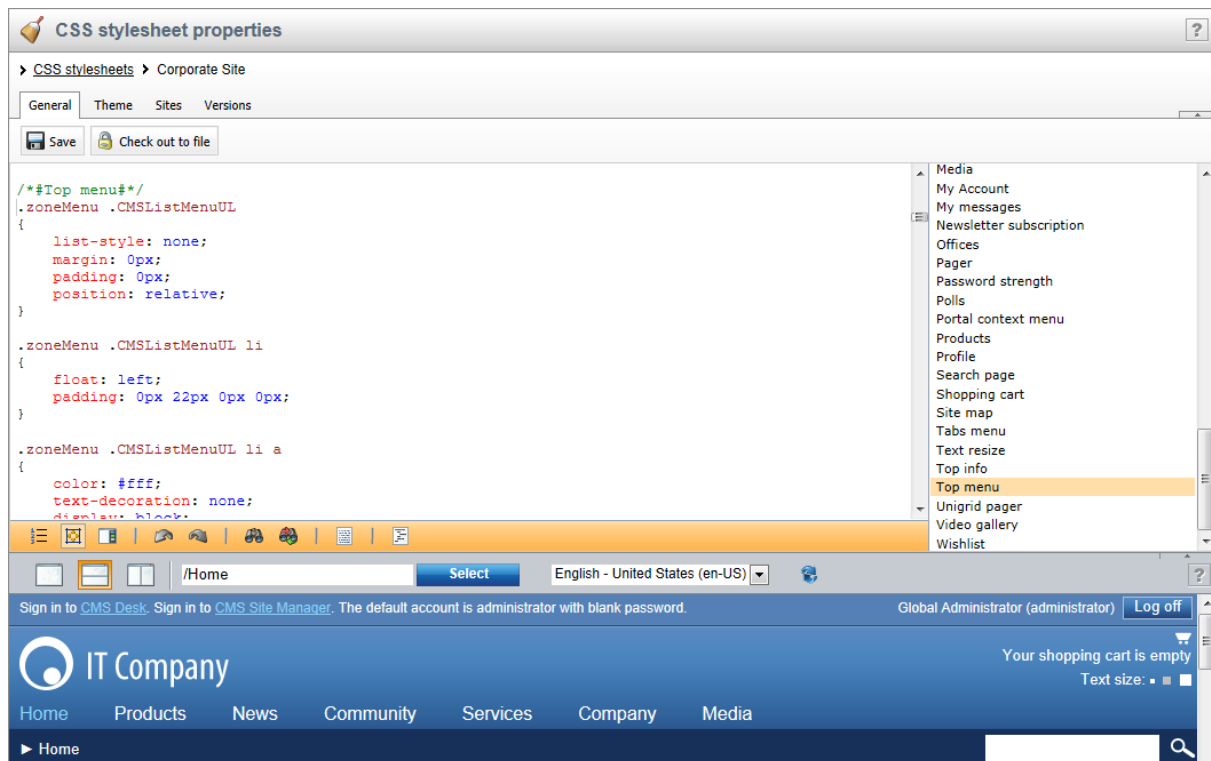
.zoneMenu .CMSListMenuUL .CMSListMenuHighlightedLIfirst a,
.zoneMenu .CMSListMenuUL .CMSListMenuLIfirst a
{
    padding-left: 0px;
}
```

...

As you can see, these are standard CSS styles. The styles can be modified in the global stylesheet of the given site.

Go to **Site Manager -> Development -> CSS stylesheets** and edit (✎) the **Corporate Site** stylesheet. Then click the  **Preview** button in the header of the **General** tab, which will allow you to view the site's pages while working with the stylesheet's code. Switch to a  **Horizontal layout** using the appropriate button on the preview toolbar. You also need to specify a page for the preview, so for example enter `/Home` into the path textbox on the toolbar and **Refresh** (🔄) the page section.

To allow easy navigation, the CSS code is separated into blocks that can be accessed using the panel on the right side of the editor. Select the **Top menu** bookmark and the editor will scroll down to the definitions of the CSS classes used to style the menu (starting with those listed above). In the page preview section below, you can see the default appearance of the website's main menu.



The screenshot shows the 'CSS stylesheet properties' window for the 'Corporate Site'. The 'General' tab is active, and the 'Top menu' bookmark is selected in the right-hand navigation panel. The main editor area contains the following CSS code:

```

/*#Top menu#*/
.zoneMenu .CMSListMenuUL
{
    list-style: none;
    margin: 0px;
    padding: 0px;
    position: relative;
}

.zoneMenu .CMSListMenuUL li
{
    float: left;
    padding: 0px 22px 0px 0px;
}

.zoneMenu .CMSListMenuUL li a
{
    color: #fff;
    text-decoration: none;
    display: block;
}

```

The preview section at the bottom shows the website's main menu with the 'Top menu' item highlighted in orange.

Now we will change the background color of highlighted menu items to orange. Scroll down in the CSS code until you find the class definition shown below and insert the highlighted line:


### [CSS]

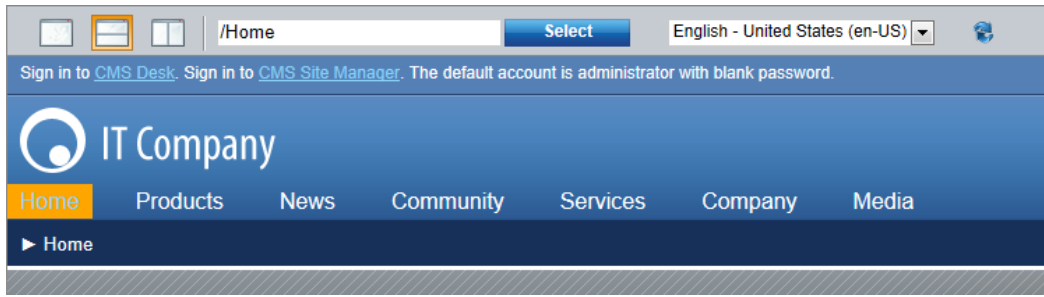
```

.zoneMenu .CMSListMenuHighlightedLI a,
.zoneMenu .CMSListMenuHighlightedLIfirst a
{
    color: #8cd2f8 !important;
}

```

```
background-color: orange;  
text-decoration: none;  
}
```

Click  **Save** to confirm the change. The page preview section will automatically be refreshed and you will be able to see the modified menu.



Of course, the new style will also be applied to the live site.


## Defining the style of a single menu item

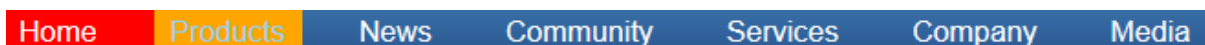
Every document may have its own style that is used when the document is displayed in a menu. We will try to modify the style of the **Home** menu item. Go to **CMS Desk -> Content** and select **Home** from the content tree. Then switch to the **Properties -> Navigation** tab. Here you can define the following settings:

- **Menu caption** - the name of the document when it's displayed in a menu.
- **Show in navigation** - indicates if the document should be displayed by navigation web parts and controls, i.e. included in menus.
- **Show in sitemap** - indicates if the document should be included in the website's sitemap.
- **Menu actions** - determines what happens when the document's menu item is clicked by a user. The link can be disabled, a JavaScript command can be executed, or a specified URL can be opened.
- **Search & SEO** - can be used to exclude the document from searches and set up advanced properties related to the website's Google sitemap.
- **Menu design** - the fields in this section allow you to set styles applied to the particular document's menu item. The styles can be specified for three different scenarios: standard, when a user hovers over the menu item, and when the given document is currently selected (highlighted).

Enter the following value into the **Menu item style** value (in the **Menu item design** section):

*background-color: red;*

Confirm the change by clicking  **Save**. To view the result, switch to **Live site** mode and select **Products** in the website's menu. The **Home** menu item will now be displayed in red:



**Part**

---

**VII**

**Creating a new site using ASPX templates**

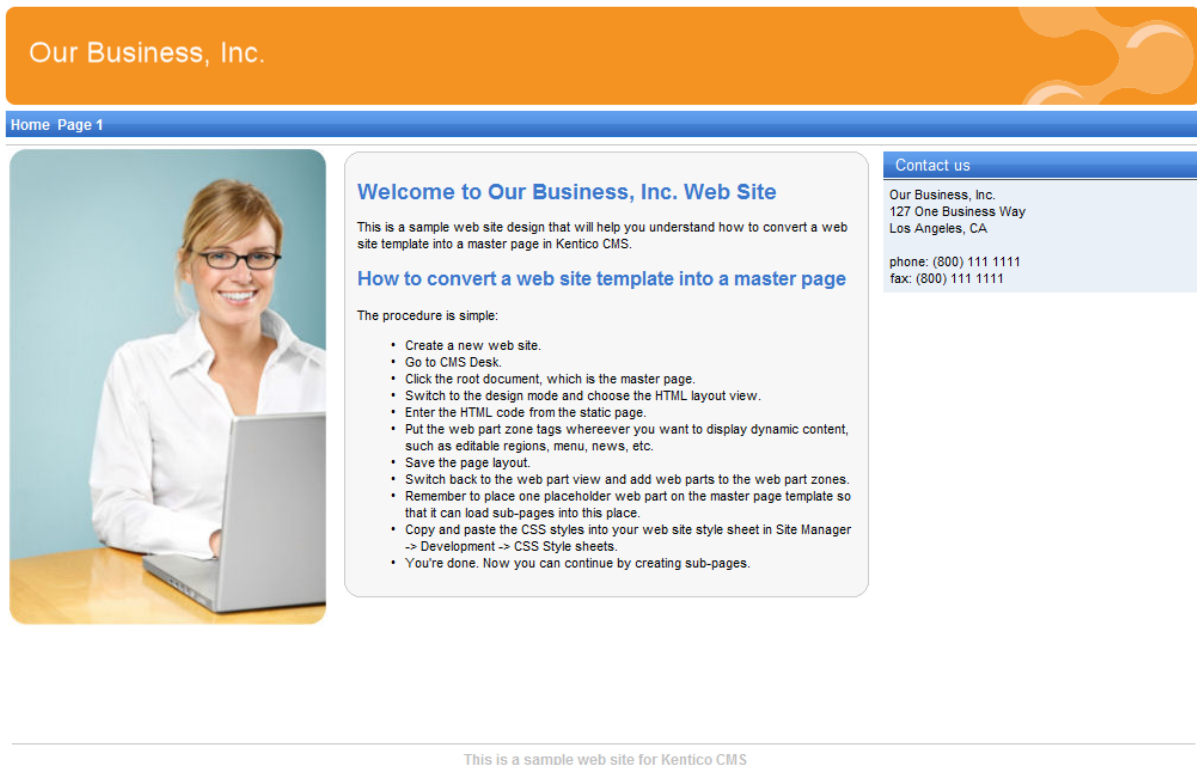
---

## 7 Creating a new site using ASPX templates

### 7.1 Overview

This chapter of the tutorial guides you through the creation of a simple website using ASPX page templates developed in Visual Studio. You will learn how to define site structure, design, how to create your own pages and page templates.

During this tutorial, we will use a static website template that is similar to what a developer gets from a graphic designer. It looks like this:



You can find the static page template in the **C:\Program Files\Kentico CMS\<version>\CodeSamples\SampleWebTemplate** folder. The template consists of the *home.htm* file, a *Styles* folder and an *app\_themes* folder with images.

### 7.2 Creating a new web site using the wizard


The following topics assume that you have previously installed a sample Corporate Site. We will leave the existing website and add a new site running under the *http://127.0.0.1* domain.

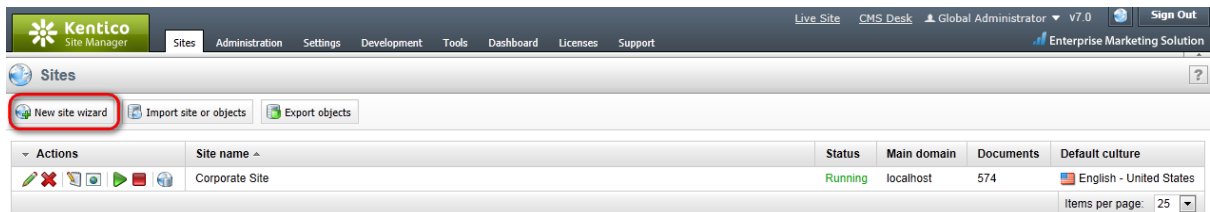


#### Multiple sites and Visual Studio's built-in web server

If you are using the **built-in web server in Visual Studio instead of IIS**, you need to **stop the CorporateSite** site in the Site Manager -> Sites dialog first and then you can

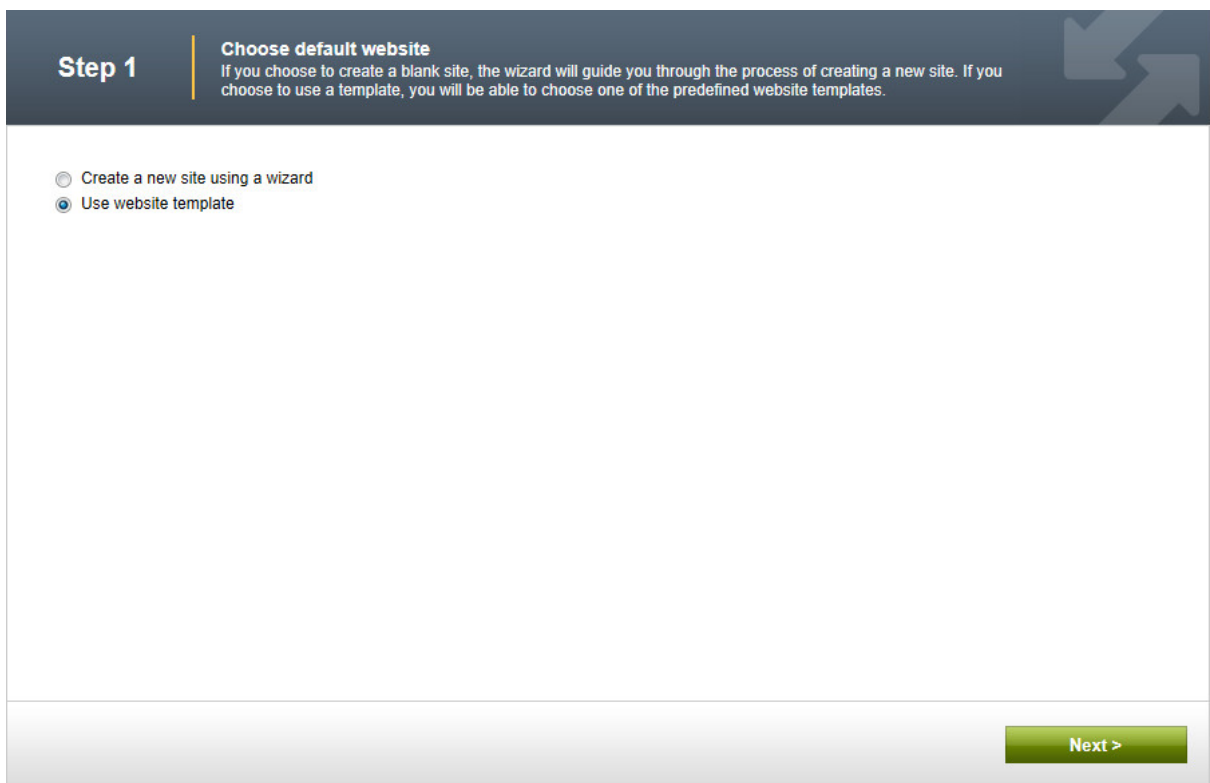
continue. Since the built-in web server doesn't support any other domain than localhost, you will use the *localhost* domain again.

1. Sign in as *Administrator* to **Site Manager** -> **Sites**.
2. Click  **New site wizard**.



The **New site wizard** opens.

3. Select **Use website template**.



Click **Next**.

4. Choose the **Blank site ASPX** website template.



**Step 2** | **Choose website template**  
Choose the predefined website template that will be used for your new website. The website template may contain site structure, design, basic content, new document types and other settings.

give you an idea of how they can be used on your web site. You can also modify this web site and use it as a base for the development of your own site. It uses the portal engine and is recommended for developers who are new to Kentico CMS.

**Intranet Portal**  
This is a ready-to-use website template for a sample intranet portal site. The site showcases many features useful for a company intranet, such as sub-sections dedicated to departments or workgroups and document or project management. You may use it as a base for your own company intranet by replacing the sample data with your own. It uses the portal engine and may be used by developers who are new to Kentico CMS.

**Blank Site**  
This template is intended for developers who want to create a new web site from scratch. It uses the portal engine.

**ASPX Blank Site ASPX**  
This template is intended for developers who want to create a new web site from scratch. It uses the ASPX page templates that require a higher level of .NET knowledge than the portal engine.

< Previous      Next >

Click **Next**.


5. Enter the following details for the website:

- **Site display name:** My website
- **Site code name:** mysite
- **Domain:** 127.0.0.1 (if you are using Visual Studio's built-in web server, set the **Domain** value to *localhost*)

### Step 3

#### Enter new site settings

Enter the display name and code name of the website. The Domain field must contain the domain that you will use to access the website during development (you may change it when the site goes live). The default culture is the main language of the website.

Site display name:  

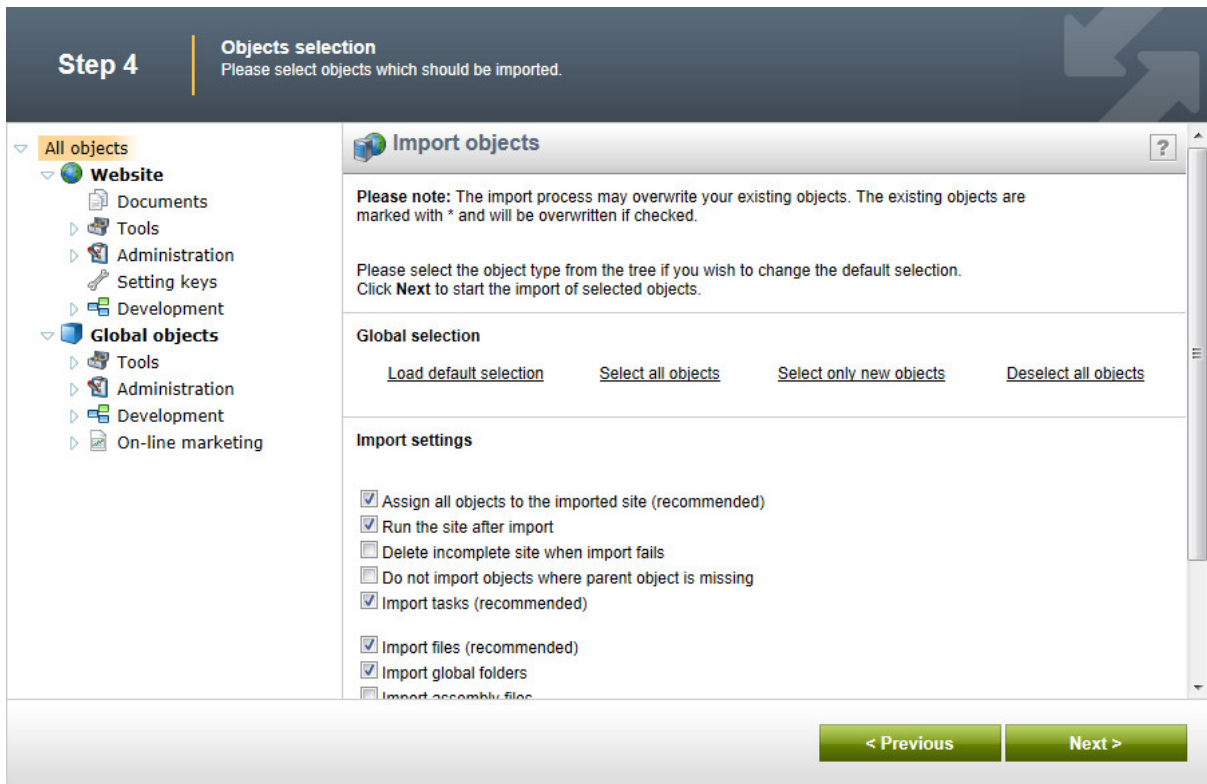
Site code name:

Domain name:

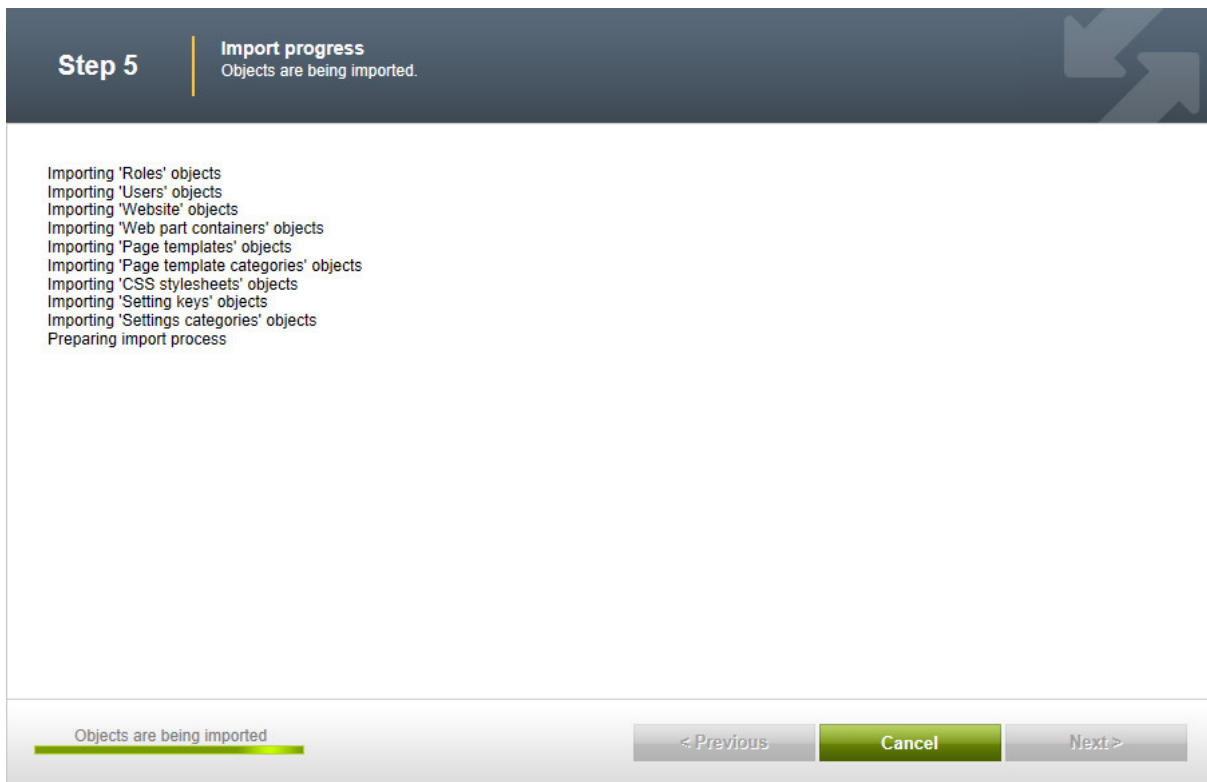
[< Previous](#) [Next >](#)

Click **Next**.

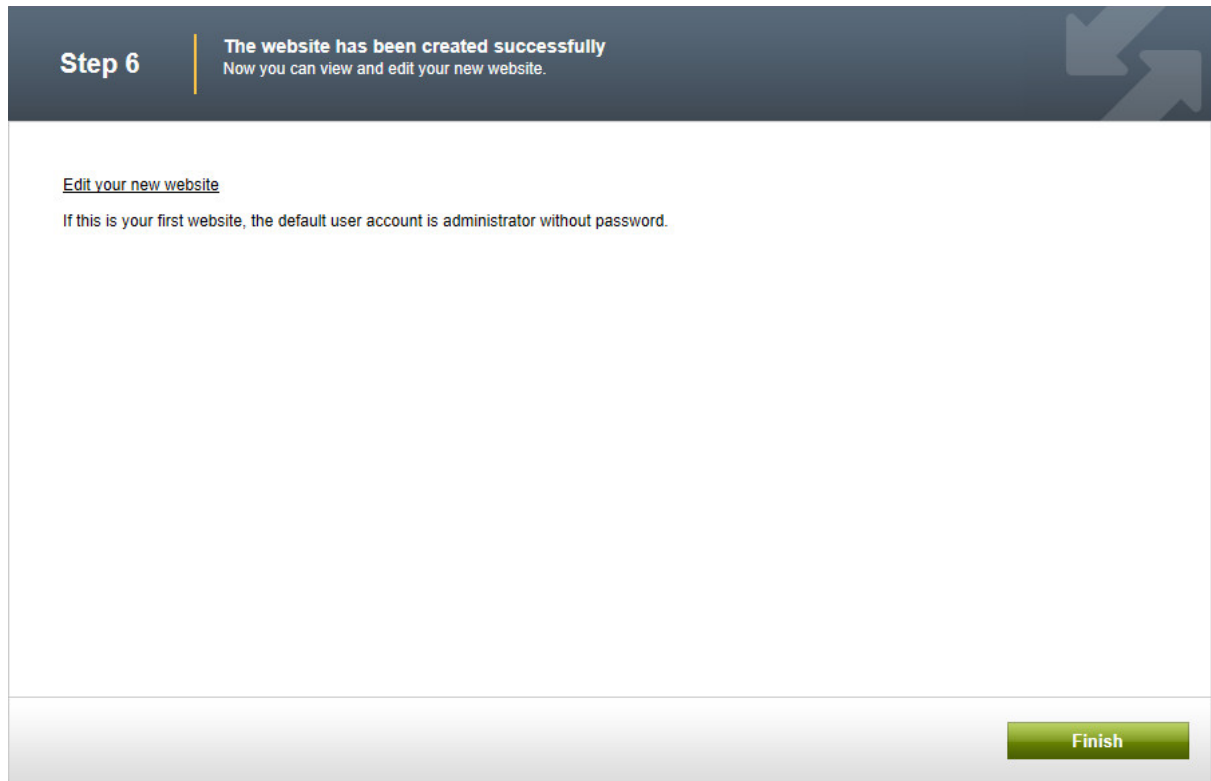
6. The fourth step of the wizard allows you to select which objects the system imports into the new site. Do not change anything and click **Next**.



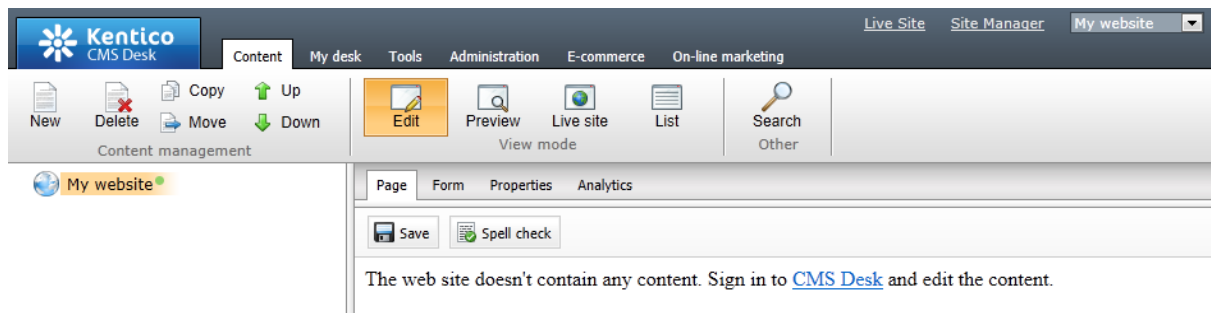
The fifth step shows the progress of the object import.



7. Click **Edit your new website** once the import is complete.



A new browser window with Kentico CMS Desk opens at domain 127.0.0.1. You need to sign in again since authentication is not shared across different domains by default. After you sign in, you will see your new, empty website:



You have created the base for your new website. The next topics describe how to implement the required design.

## 7.3 Creating the CSS stylesheet

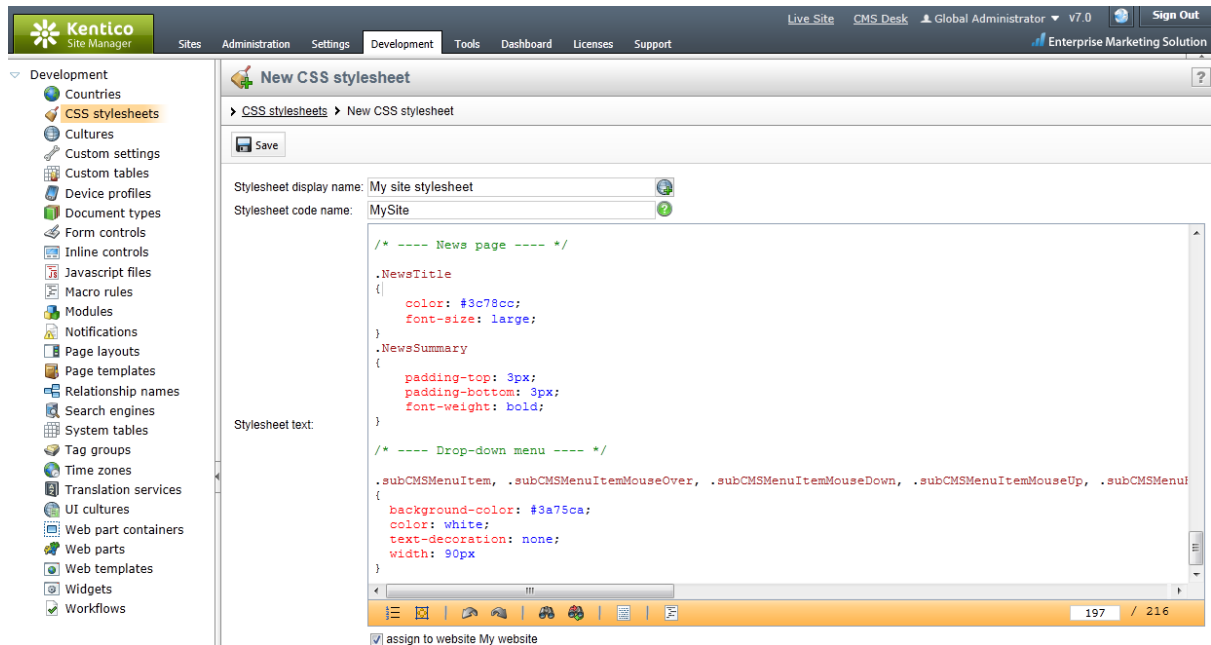
Before you start editing your new website, prepare a new CSS stylesheet based on the styles and images of the sample website template.

1. Go to **Site Manager -> Development -> CSS Stylesheets**.

2. Click  **New CSS stylesheet**.

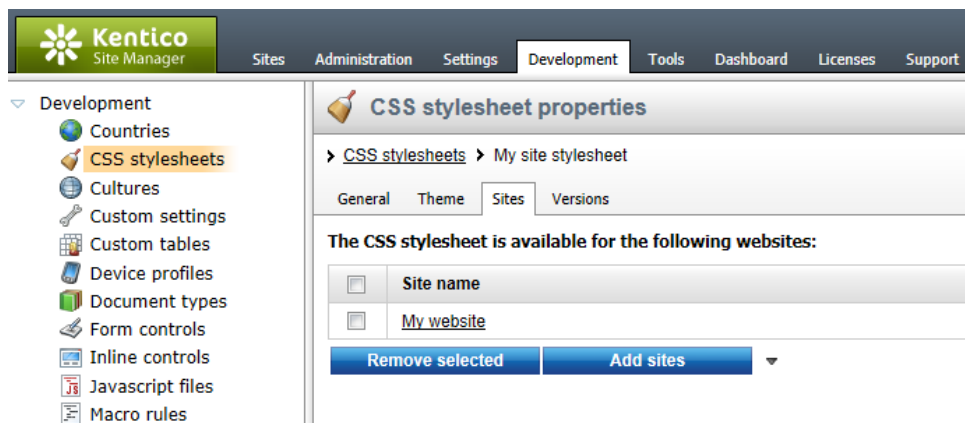
3. Enter the following values:

- **Stylesheet display name:** My site stylesheet
- **Stylesheet code name:** MySite
- **Stylesheet text:** copy and paste all code from *C:\Program Files\Kentico CMS\<version>\CodeSamples\SampleWebTemplate\Styles\main.css*



4. Click  **Save**.

5. Switch to the **Sites** tab and assign the stylesheet to **My website**.



6. Navigate to **Site Manager -> Sites** and **Edit** () the properties of **My website**.

7. On the **General** tab, select *My site stylesheet* as the **Site CSS stylesheet**.

The screenshot shows the 'Site properties' page in the Kentico Site Manager. The page is titled 'Site properties' and is under the 'Sites' section for 'My website'. There are tabs for 'General', 'Domain aliases', 'Cultures', 'Off-line mode', and 'Site objects'. A 'Save' button is visible. The form contains the following fields:

- Site display name: My website
- Site code name: mysite
- Site domain name: 127.0.0.1
- Default content culture: English - United States (with a 'Change' button)
- Visitor culture: (Automatic)
- Site CSS stylesheet: My site stylesheet (with 'Edit' and 'New' buttons)
- Editor CSS stylesheet: (site stylesheet) (with 'Edit' and 'New' buttons)
- Site description: Blank web site using ASPX templates

8. Click **Save**. This ensures that all pages of your new website load the appropriate stylesheet.

9. Copy the **SampleWebTemplate\app\_themes\MySite** folder to the **App\_Themes** folder under your web project.

The folder contains graphics for this website template. This location ensures that the images are exported as part of the website if you decide to move the website in the future. Note that the folder under **App\_Themes** must have the same name as the code name of the CSS stylesheet: **MySite**



### CSS stylesheet URL and relative paths

We have adjusted the image paths in the sample CSS stylesheet so that they match the target folders in your new website. In real-world scenarios, you will need to adjust the paths manually. **The URLs of images in the CSS stylesheets are always relative to the location of the web project.**

The URL of the CSS stylesheet is:

```
<web project>/CMSPages/GetResource.ashx?stylesheetname=MySite
```

which means, you need to link to files in the app\_themes folder like in the example below:

```
../app_themes/mysite/images/imagename.gif
```

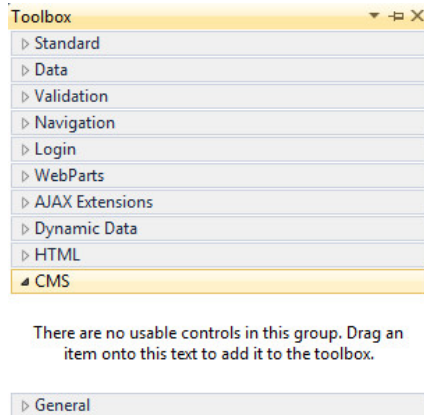
## 7.4 Opening and configuring the web project

Open your Kentico CMS web project in Visual Studio. You can open it either through the **WebProject.sln** file or using the **File -> Open Web Site** option in the menu.

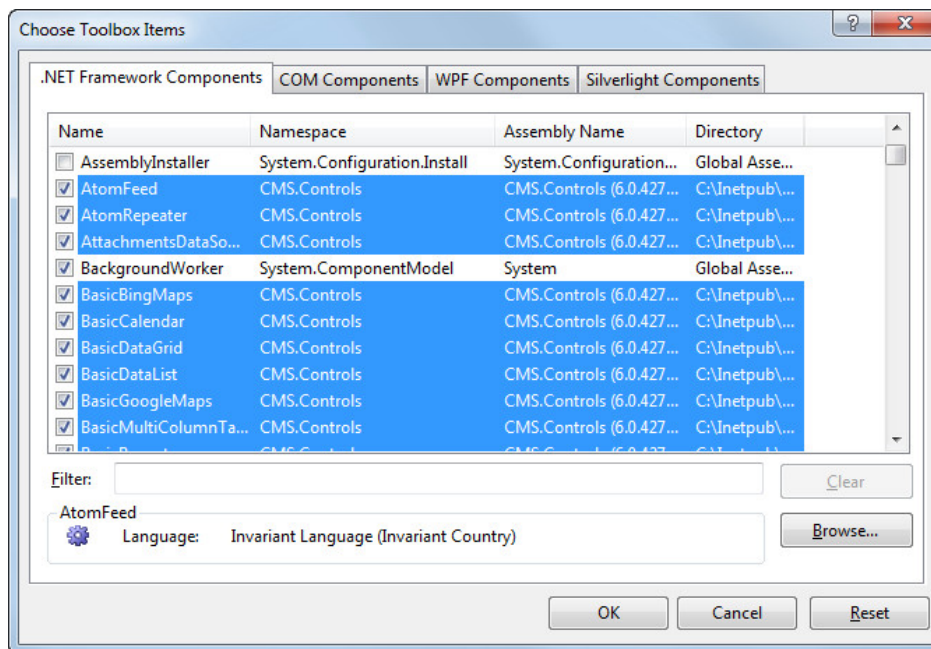
### Adding Kentico CMS controls to the Visual Studio Toolbox

To make it easier to work with Kentico CMS components on your ASPX pages, add the built-in set of controls to your Visual Studio Toolbox.

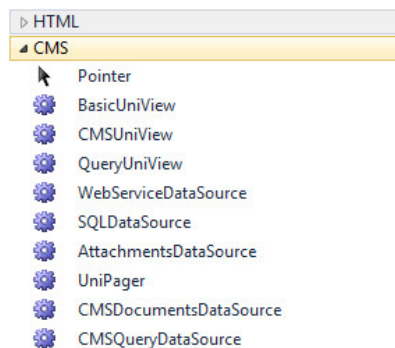
1. Edit any ASPX web form file, for example *Default.aspx* under the project root. This is necessary, because the toolbox only offers the controls when working with ASPX markup.
2. Right-click the **Toolbox** and choose **Add Tab**.
3. Type the name of the new tab (e.g. CMS) and press Enter:



4. Right-click the new tab and select **Choose items...** from the context menu.
5. In the **Choose Toolbox Items** dialog, click **Browse** and locate the **CMS.Controls.DLL** library in the **bin** folder under your website. Click **Open** and then click **OK**.



The controls are now added to the Toolbox. You can drag and drop the controls onto your Web forms.



## 7.5 Master page

Now we will create a master page for the website containing a header, navigation menu and footer. This master page will be shared by all ASPX templates used to build the site's pages.

1. Open your web project in Visual Studio, right-click the **CMSTemplates** folder in the Solution Explorer window and select **New Folder**. Name the folder *MySite*.

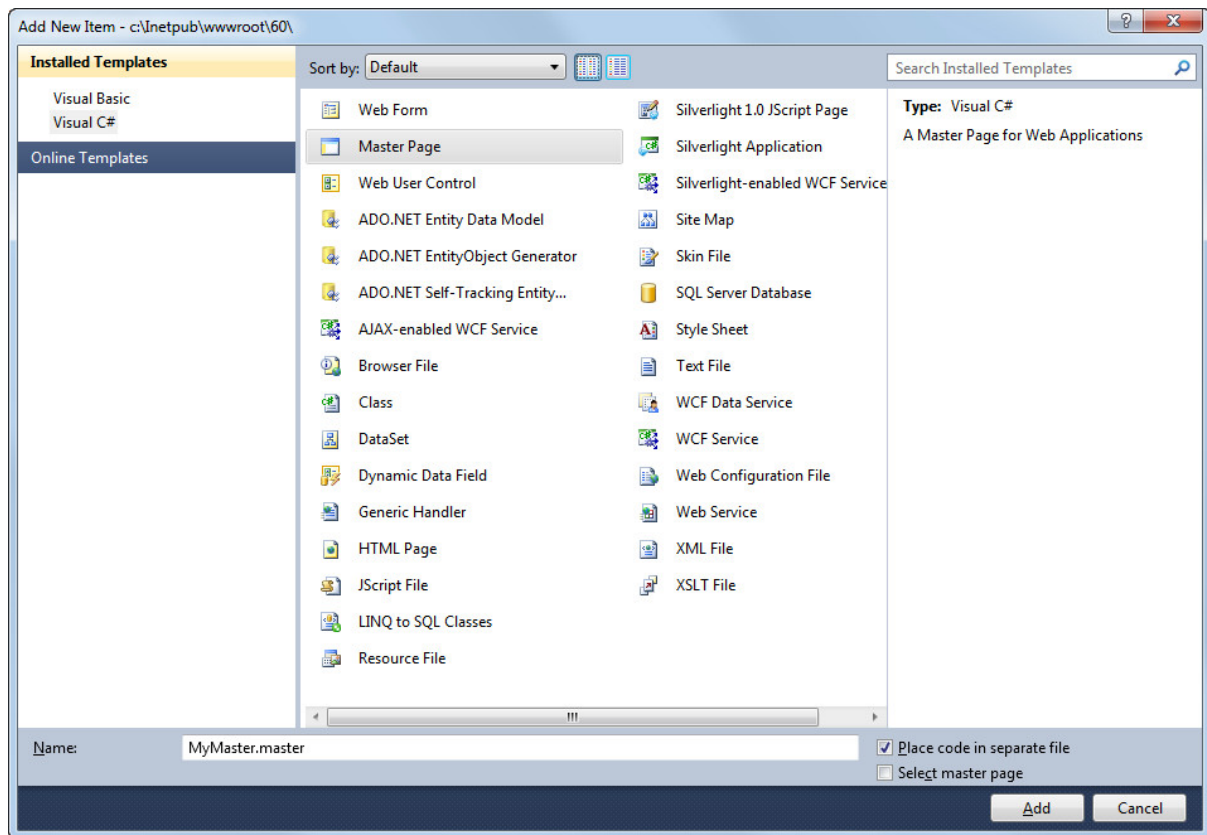


### Folder name

We recommend using a folder name that matches the code name of your site. This ensures that the system exports/imports the folder's content along with the website when you deploy it to another instance of Kentico CMS.



2. Right-click the **MySite** folder, select **Add new item...** and create a new **Master page** called *MyMaster.master*.



3. Delete all default ASPX code of the master page (in the **Source** view) except for the first line with the **<%@ Master %>** directive and add the following code instead:

```
<%=DocType%>
<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
  <title id="Title1" runat="server">My website</title>
  <asp:literal runat="server" id="lt1Tags" enableviewstate="false" />
</head>
<body class="<%=BodyClass%>" <%=BodyParameters%>>
  <form id="form1" runat="server">
    <ajaxToolkit:ToolkitScriptManager ID="manScript" runat="server"
    EnableViewState="false" ScriptMode="Release" />
    <cms:CMSPortalManager ID="CMSPortalManager1" runat="server" />
  </form>
</body>
</html>
```

- The **ToolkitScriptManager** control allows AJAX components to work on the pages of your site (required).
- The **CMSPortalManager** control ensures the loading and saving of content between the database

and editable regions. It also provides the management necessary for web part or widget zones defined on child ASPX pages.

4. Open the sample **home.htm** file (in *C:\Program Files\Kentico CMS\<version>\CodeSamples\SampleWebTemplate*) and copy the HTML code from inside the **<body>...</body>** tags. Paste this code into the body of the master page after the **<cms:PortalManager>** control.

5. Delete all code in the **<!-- main content --> ... <!-- /main content -->** section and replace it with the following control instead:

```
<asp:ContentPlaceHolder ID="plcMain" runat="server"></asp:ContentPlaceHolder>
```

Because you are creating a master page, you do not need the actual content of the Home page, only the logo, main menu and footer. The replacement code adds a standard ASP.NET control that ensures the loading of pages inside the master page.

The code of the master page's **<body>** element should now look like this:

```
<body class="<%=BodyClass%>" <%=BodyParameters%>">
  <form id="form1" runat="server">
    <ajaxToolkit:ToolkitScriptManager ID="manScript" runat="server"
    EnableViewState="false" ScriptMode="Release" />
    <cms:CMSPortalManager ID="CMSPortalManager1" runat="server" />

    <div class="MainDiv">

      <!-- logo -->
      <br />
      <div class="Logo">
        &nbsp;
      </div>

      <!-- main menu -->
      <div class="MainMenu">
        <table cellspacing="2" cellpadding="2" border="0">
          <tr>
            <td class="MainCMSMenuHighlightedMenuItem">Home</td>
            <td class="MainCMSMenuItem">Page 1</td>
          </tr>
        </table>
      </div>

      <!-- main content -->
      <asp:ContentPlaceHolder ID="plcMain" runat="server"></asp:
      ContentPlaceHolder>
      <!-- /main content -->

      <!-- footer -->
      <div class="Footer">
        This is a sample web site for Kentico CMS
      </div>

    </div>

  </form>
</body>
```

6. Switch to code behind of the master page (*MyMaster.master.cs*) and add a reference to the **CMS.UIControls** namespace:

[C#]

```
using CMS.UIControls;
```

[VB.NET]

```
Imports CMS.UIControls
```

7. Change the class definition so that the master page inherits from the **TemplateMasterPage** class:

**[C#]**

```
public partial class CMSTemplates_MySite_MyMaster : TemplateMasterPage
```

**[VB.NET]**

```
Partial Class CMSTemplates_MySite_MyMasterVB  
    Inherits TemplateMasterPage
```

8. Override the **CreateChildControls** method in the class according to the following code:

**[C#]**

```
protected override void CreateChildControls()  
{  
    base.CreateChildControls();  
    PageManager = CMSPortalManager1;  
}
```

**[VB.NET]**

```
Protected Overrides Sub CreateChildControls()  
    MyBase.CreateChildControls()  
    PageManager = CMSPortalManager1  
End Sub
```

9. Add an override for the **OnPreRender** method:

**[C#]**

```
protected override void OnPreRender(EventArgs e)  
{  
    base.OnPreRender(e);  
    this.ltlTags.Text = this.HeaderTags;  
}
```

**[VB.NET]**

```
Protected Overrides Sub OnPreRender(e As EventArgs)  
    MyBase.OnPreRender(e)  
    Me.ltlTags.Text = Me.HeaderTags  
End Sub
```

10. Save both master page files.

Continue editing the master page according to the instructions in the [Main menu](#) topic.

## 7.6 Main menu

Now we will add a dynamic **drop-down menu** to the [master page](#). You can implement the drop-down menu using either the *CMSMenu* or *CMSListMenu* control. The example uses the first option, which is easier to understand if you are not familiar with advanced CSS styles.



### Note

If you prefer a drop-down menu based on CSS styles and UL/LI elements, you can try using the *CMSListMenu* later. See the [Kentico CMS Controls Reference](#) for additional details and examples.

1. Edit the **MyMaster.master** file in Visual Studio.
2. Remove the `<table>` element used as a static menu inside the `<div class="MainMenu">` element. Instead, drag the **CMSMenu** control from the toolbox to this location.
3. Set the following properties of the CMSMenu control (you can find them in the *Behavior* section of the Visual Studio *Properties* window):

Property	Value	Description
<b>Path</b>	/%	Configures the menu to display pages starting from the root of the site structure.
<b>Layout</b>	Horizontal	Sets a horizontal layout for the menu.
<b>CSSPrefix</b>	;sub	Allows you to add prefixes before the names of the CSS classes applied to the menu. The <i>;sub</i> value uses unmodified class names for the main (first) menu level and the <b>sub</b> prefix for the second level and all other sub-levels.
<b>Cursor</b>	Pointer	Specifies the type of mouse cursor displayed when a user hovers over the menu.

The code of the main menu section should now look like this:

```
<!-- main menu -->
<div class="MainMenu">
  <cms:CMSMenu ID="CMSMenu1" runat="server" Path="/%" Layout="Horizontal"
  CSSPrefix=";sub" Cursor="Pointer" />
</div>
```

4. **Save** the changes.

The master page is now prepared and you can assign it to the site's ASPX templates. Continue by creating the [Home page](#) of the website.



### Kentico CMS Controls and Web Parts

Kentico CMS provides a set of flexible server controls in the **CMS.Controls.dll** library, but large amounts of the built-in functionality are only available through web parts stored in the **CMSWebParts** folder.

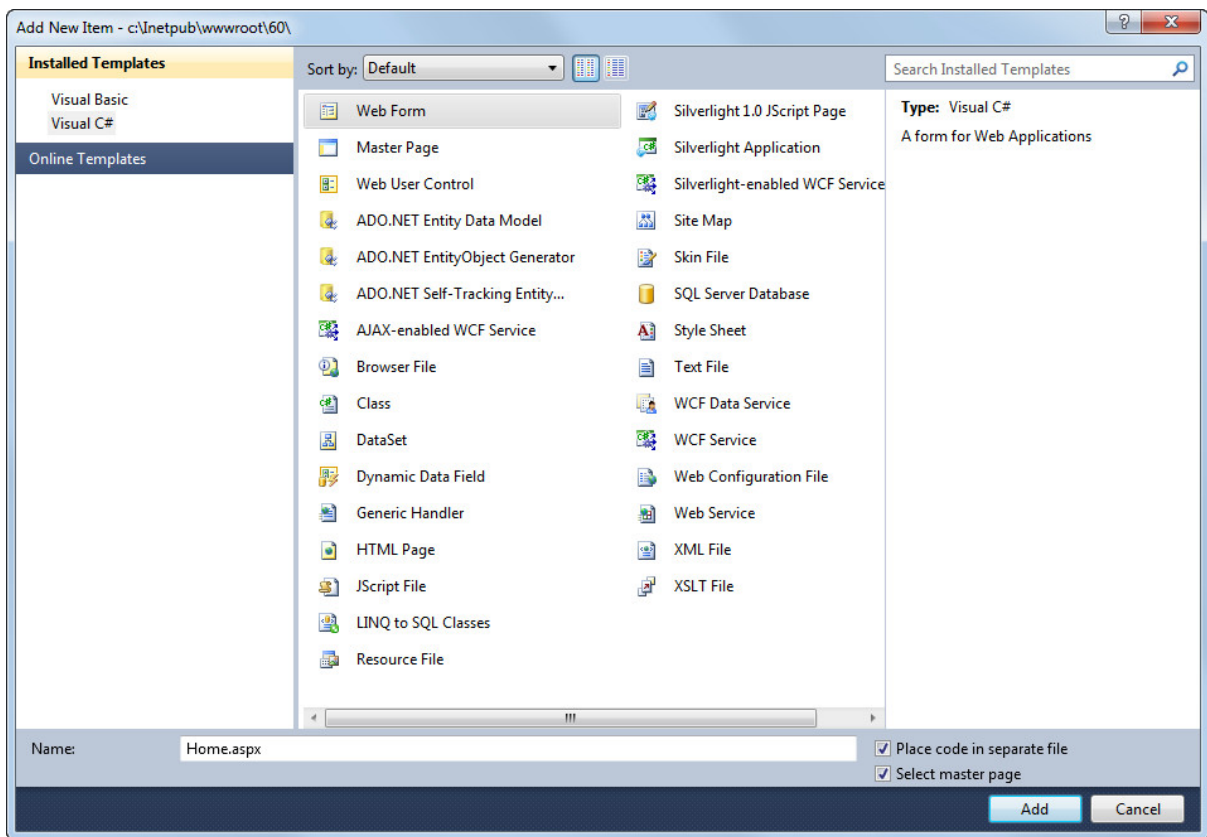
These web parts are standard ASCX user controls and you can use them on both portal engine templates and ASPX pages. To add a web part onto your ASPX pages, drag it from the *Solution explorer* and set the properties in the *Properties* window.

## 7.7 Home page

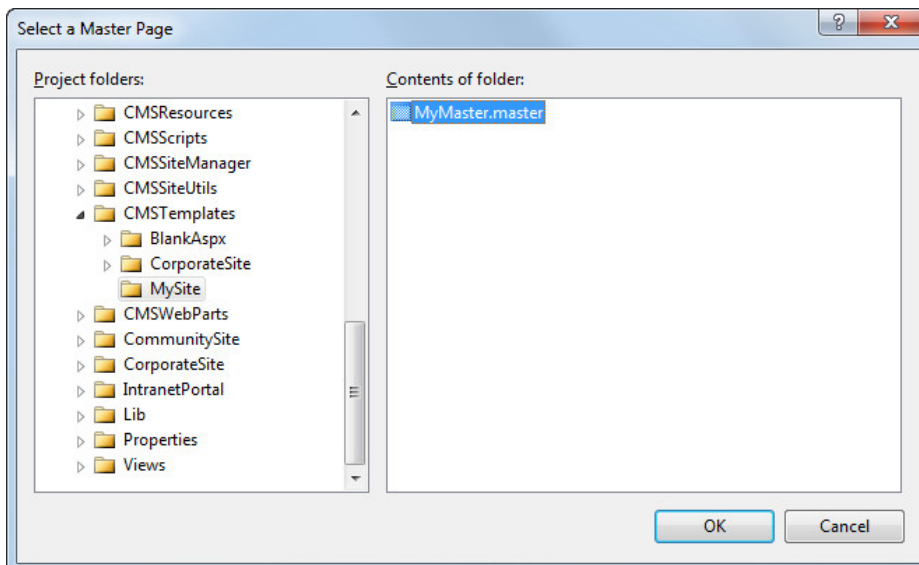
This topic describes how to create the home page of the website.

### Preparing the ASPX source file

1. Edit your web project in Visual Studio, right-click the **CMSTemplates/MySite** folder in the **Solution Explorer** and click **Add new item**.
2. Create a **Web Form** named *Home.aspx* and check the **Select master page** box.



3. Click **Add** and choose the **MyMaster.master** page from the **CMSTemplates/MySite** folder.



4. Open the sample **home.htm** file (in *C:\Program Files\Kentico CMS\<version>\CodeSamples\SampleWebTemplate*) and copy the HTML code from inside the **<!-- main content -->** section. Paste this code inside the **<asp:Content>** element of the *Home.aspx* file.

5. Remove the static text content from the page:

- The "Welcome to Our Business, Inc. Web Site..." text inside the table in the **<!-- center box -->** section
- The "Our Business, Inc. ..." text in the **<!-- right column -->** section

The content of the web form should now match the following:

```
<asp:Content ID="Content1" ContentPlaceHolderID="plcMain" Runat="Server">

<!-- main content -->
  <table style="width:100%;height:500px;border: 0px">
    <tr valign="top">
      <!-- left column -->
      <td style="width:280px" class="HomePageLeftColumn">
      </td>
      <!-- center column -->
      <td style="padding: 3px 5px 0px 5px;width:450px;">
        <!-- center box -->
        <table cellpadding="0" cellspacing="0" border="0"
class="ContainerWithCorners" width="100%">
          <tr class="ContainerWithCornersRow">
            <td class="ContainerWithCornersTopLeft">&nbsp;</td>
            <td class="ContainerWithCornersTop">&nbsp;</td>
            <td class="ContainerWithCornersTopRight">&nbsp;</td>
          </tr>
          <tr>
            <td class="ContainerWithCornersLeft">&nbsp;</td>
            <td class="ContainerWithCornersContent" valign="top">

            </td>
            <td class="ContainerWithCornersRight">&nbsp;</td>
          </tr>
          <tr class="ContainerWithCornersRow">
            <td class="ContainerWithCornersBottomLeft">&nbsp;</td>
            <td class="ContainerWithCornersBottom"></td>
            <td class="ContainerWithCornersBottomRight">&nbsp;</td>
          </tr>
        </table>
      </td>
      <!-- right column -->
      <td style="padding: 3px 0px 0px 5px;width:270px">
        <!-- text box -->
        <table cellpadding="0" cellspacing="0" style="width: 100%;
margin-bottom: 10px;" class="Blue">
          <tr>
            <td class="BoxTitle">Contact us
            </td>
          </tr>
          <tr>
            <td class="BoxArea">

            </td>
          </tr>
        </table>
      </td>
    </tr>
  </table>
</asp:Content>
```



```

        </table>
<!-- /main content -->

</asp:Content>

```

We will replace the text with editable regions so that content editors can manage the page through the CMS Desk interface.



### Using CSS-based layout instead of tables

If you prefer using a CSS-based layout, you can modify the HTML code here and replace the tables with other elements (<div>, <span>, etc.). The examples use a table-based layout by default since it is easier to understand.

6. Modify the code of the table in the **<!-- center box -->** section according to the following:

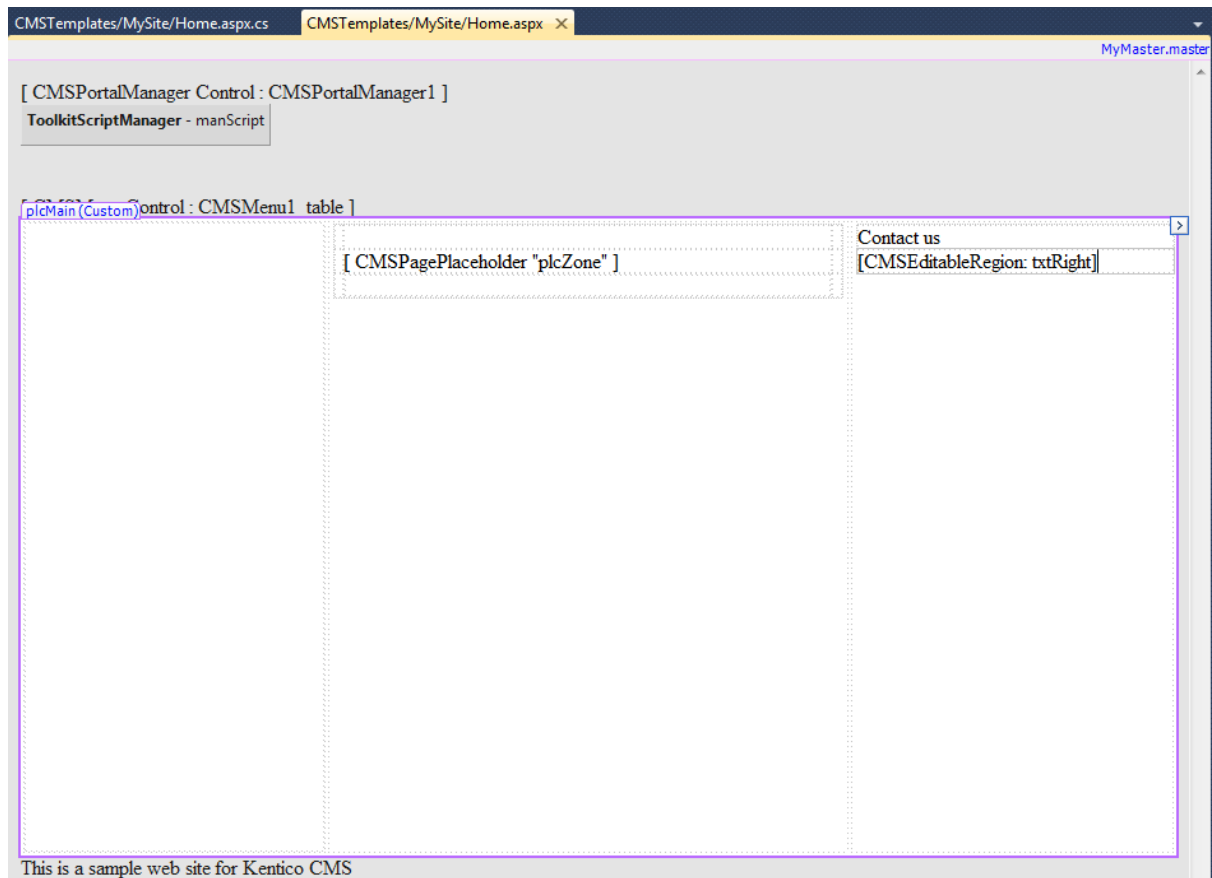
```

<!-- center box -->
<table cellspacing="0" cellpadding="0" border="0" class="ContainerWithCorners"
width="100%">
  <tr class="ContainerWithCornersRow">
    <td class="ContainerWithCornersTopLeft">&nbsp;</td>
    <td class="ContainerWithCornersTop">&nbsp;</td>
    <td class="ContainerWithCornersTopRight">&nbsp;</td>
  </tr>
  <tr>
    <td class="ContainerWithCornersLeft">&nbsp;</td>
    <td class="ContainerWithCornersContent" valign="top">
      <cms:CMSPagePlaceholder ID="plcZone" runat="server">
        <LayoutTemplate>
          <cms:CMSWebPartZone ID="zoneMain" runat="server" />
        </LayoutTemplate>
      </cms:CMSPagePlaceholder>
    </td>
    <td class="ContainerWithCornersRight">&nbsp;</td>
  </tr>
  <tr class="ContainerWithCornersRow">
    <td class="ContainerWithCornersBottomLeft">&nbsp;</td>
    <td class="ContainerWithCornersBottom"></td>
    <td class="ContainerWithCornersBottomRight">&nbsp;</td>
  </tr>
</table>

```

The **CMSPagePlaceholder** control (added to the center cell of the middle row) defines an area of the page that users can modify through their browser. Later, we will configure this area to allow content editors to customize the design of the Home page.

7. Switch to the **Design** tab and drag a **CMSEditableRegion** control from the toolbox into the bottom cell of the table on the right of the page.



8. Set the following properties of the **CMSEditableRegion** control:

- **ID:** txtRight
- **DialogHeight:** 280
- **RegionType:** HtmlEditor
- **RegionTitle:** Right content

9. Switch to the code behind of the home page (*Home.aspx.cs*) and add a reference to the **CMS.UIControls** namespace:

**[C#]**

```
using CMS.UIControls;
```

**[VB.NET]**

```
Imports CMS.UIControls
```

10. Change the class definition so that it inherits from the **TemplatePage** class:

**[C#]**

```
public partial class CMSTemplates_MySite_Home : TemplatePage
```





### [VB.NET]

```
Partial Class CMSTemplates_MySite_Home  
    Inherits TemplatePage
```

11. Save the home page files.

## Registering the page template in the system

The source files of the home page are ready. Now you need to register the home page template in Kentico CMS.

1. Open Kentico CMS in a web browser and go to **Site Manager -> Development -> Page templates**.
2. Click  **New category** and type *My website* into the **Category display name** field. Click  **Save**.
3. Click  **New template** and type *Home page* into the **Template display name** field. Click  **Save**.
4. Set the following values on the **General** tab:
  - **Template type**: ASPX + Portal page
  - **File name**: ~/CMSTemplates/MySite/Home.aspx


The screenshot shows the 'Page templates' management interface. On the left, a tree view lists various template categories, with 'My website' > 'Home page' highlighted. The main panel displays the configuration for the selected template, including fields for 'Template display name', 'Template code name', 'Category', 'Template description', and 'Thumbnail'. The 'Save' button is located at the top left of the main configuration area.

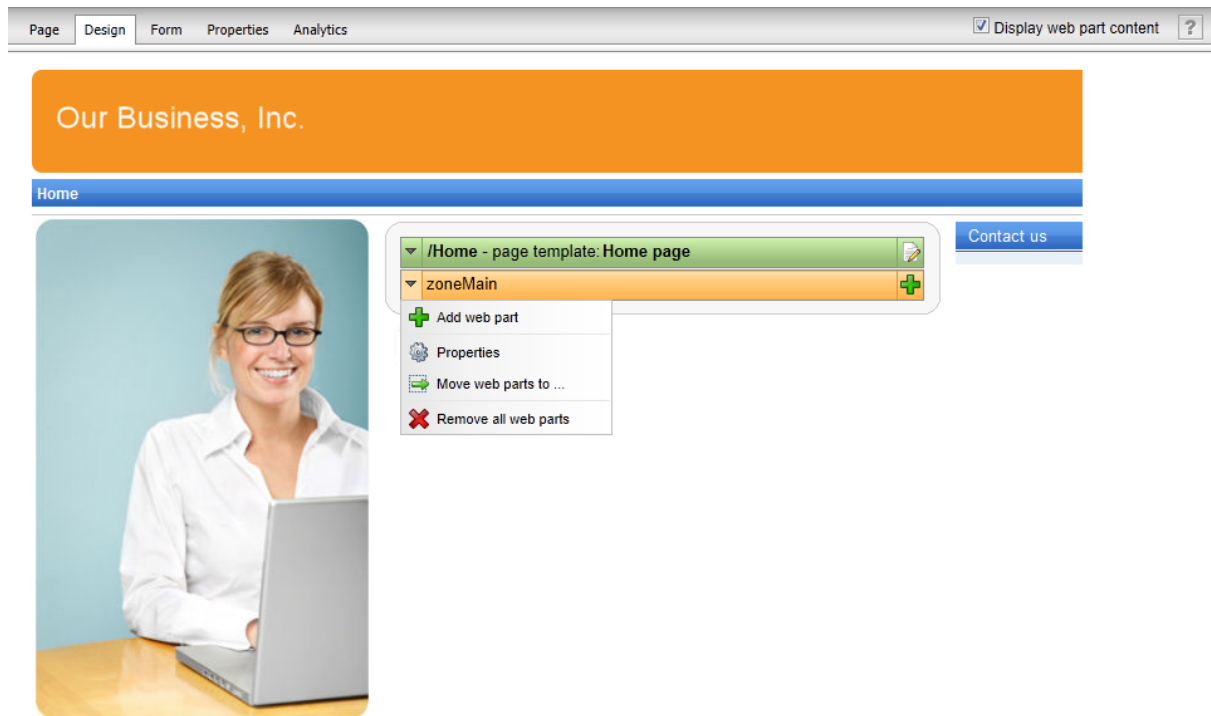
5. Click **Save**.
6. Switch to the **Sites** tab and assign the page template to **My website**.

The screenshot shows the 'Sites' tab of the 'Page templates' interface. It displays a list of websites where the selected template is available. The 'My website' site is selected, and the 'Add sites' button is visible at the bottom.

## Adding the Home page in CMS Desk

1. Go to **CMS Desk -> Content**. Select the root of the content tree and click **New**.
2. Choose the **Page (menu item)** document type.
3. Type in *Home* as the **Page name** and choose the **Use existing page template** option. Select the **My website** category and the **Home page** template.
4. Click **Save** to create the page.
5. Switch to the page's **Design** tab. Here you can see the editable area that you defined in the code of the page template.

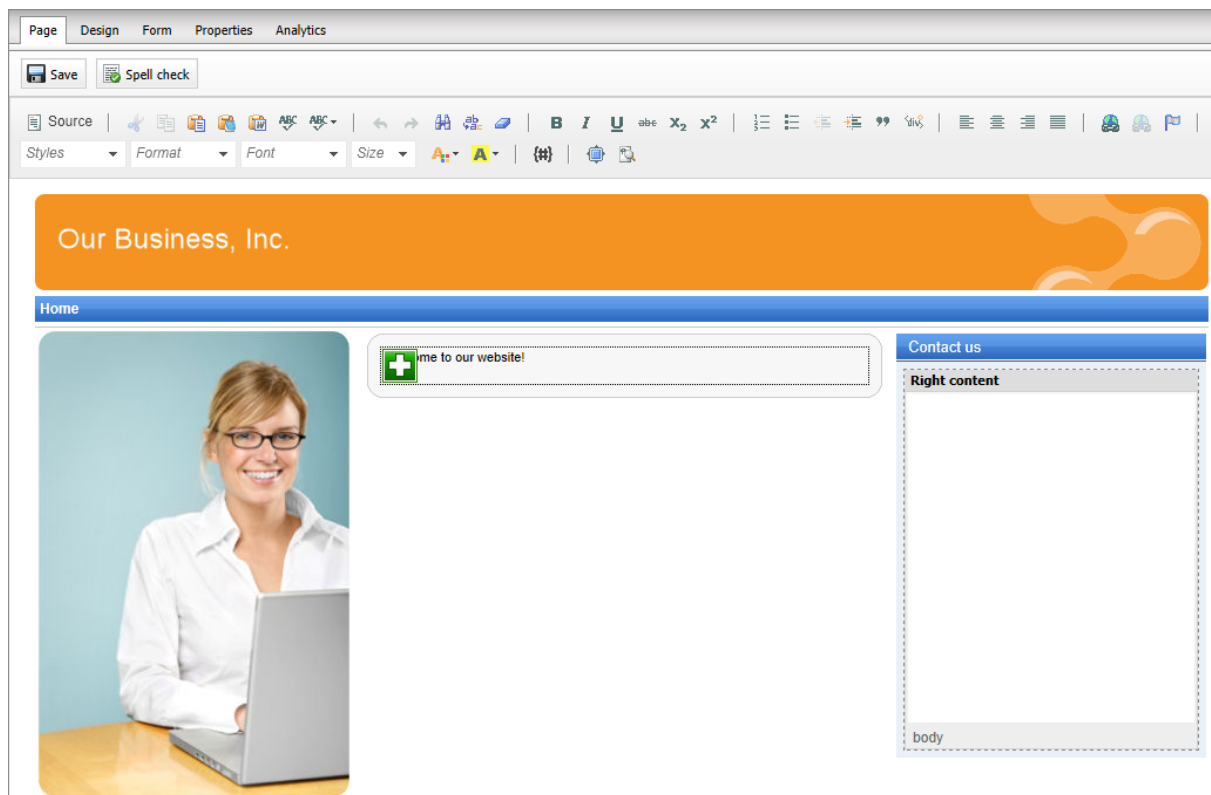
6. Right-click the header of the **zoneMain** zone and select  **Properties** to open its configuration dialog.



7. Switch the value of the **Widget zone type** property from *None* to *Customization by page editor* and click **OK**.

8. Open the **Page** tab and click **Add widget** ().

- a. Select the **General -> Text** widget.
- b. Click **OK**.
- c. Type *Welcome to our website!* into the **Text** property
- d. Click **OK** again.



You can modify the design of the page directly through the browser by adding and configuring widgets. This approach can be useful once the website has some more content or features to be displayed.

9. Type the following text into the **Right content** editable region: *Call 800 111 2222*

10. Click  **Save**.

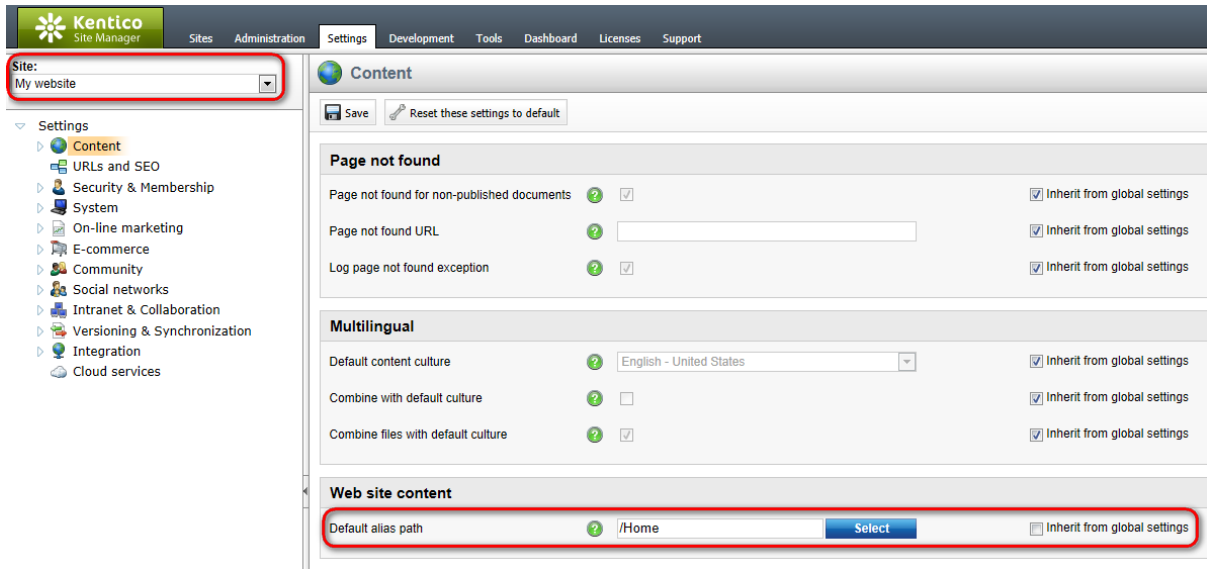
Switch to **Live site** mode to see how the home page of your website appears to visitors.

## Configuring the website's home page

When a visitor arrives on the root URL of the website (i.e. its domain name, for example *http://www.example.com*), the system needs to know which page to display as the home page.

To set the path of the website's default home page:

1. Go to **Site Manager -> Settings** and click the **Content** category in the settings tree.
2. Select **My website** in the **Site** drop-down menu.
3. Uncheck the **Inherit from global settings** box next to the **Default alias path** setting and type in */Home*, which is the alias path of your new home page.



4. Click **Save**.

When visitors access the website without specifying the URL of a particular page, the system automatically displays the Home page.

## 7.8 News page

Now we will create the News section of the website.

### Preparing the ASPX source file

1. Edit your web project in Visual Studio, right-click the **CMSTemplates/MySite** folder in the **Solution Explorer** and click **Add new item**.
2. Create a **Web Form** named *NewsPage.aspx* and check the **Select master page** box.
3. Click **Add** and choose the **MyMaster.master** page from the **CMSTemplates/MySite** folder.
4. Drag the following controls inside the `<asp:Content>` element of the news page:
  - **CMSBreadCrumbs**
  - **CMSRepeater**
5. Set the properties of the **CMSRepeater** control according to the table below (you can find them in the *Behavior* section of the Visual Studio *Properties* window):

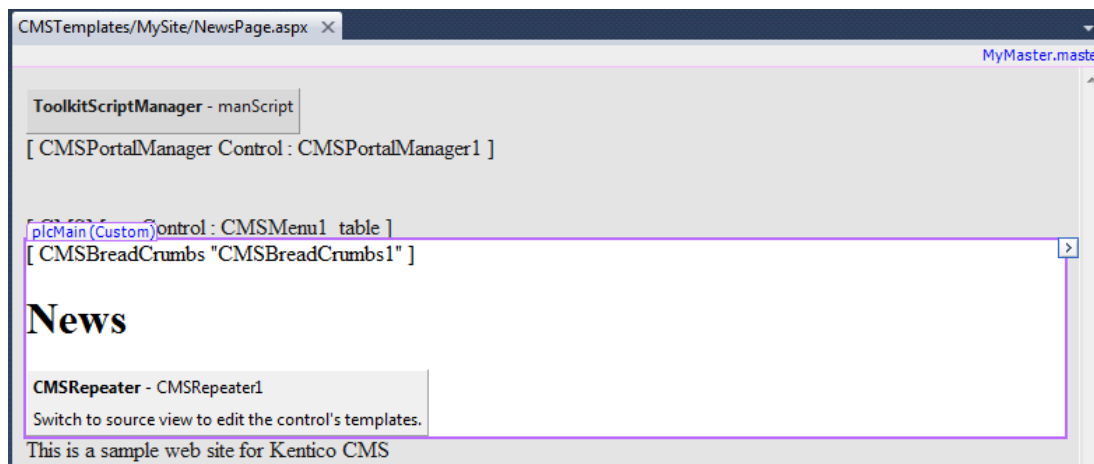
Property	Value	Description
ClassNames	cms.news	Configures the repeater to display only documents of the <i>cms.news</i> type.
TransformationName	cms.news.pre view	Assigns the transformation that the repeater uses to display the list of news items.

SelectedItemTransformationName	cms.news.default	When a user selects a specific news item on the website, the repeater displays the details according to the specified transformation.
ItemSeparator	<hr />	Defines the HTML code placed between individual news items in the list.

6. Add the following HTML code between the two controls:

```
<h1>News</h1>
```

If you view the web form on the **Design** tab, you should see the structure of the page as shown below:



7. Switch to the code behind of the news page (*NewsPage.aspx.cs*) and add a reference to the **CMS.UIControls** namespace:

**[C#]**

```
using CMS.UIControls;
```

**[VB.NET]**

```
Imports CMS.UIControls
```

8. Change the class definition so that it inherits from the **TemplatePage** class:

**[C#]**

```
public partial class CMSTemplates_MySite_NewsPage : TemplatePage
```

**[VB.NET]**






```
Partial Class CMSTemplates_MySite_NewsPage
    Inherits TemplatePage
```

9. Save the news page files.

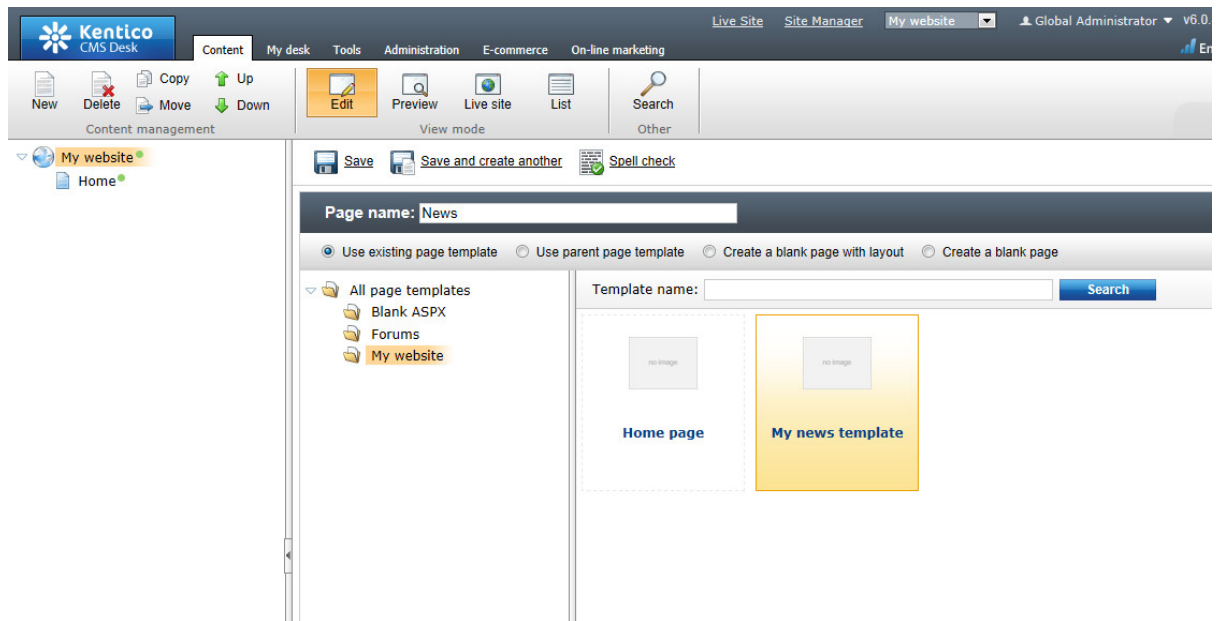
## Registering the page template




The source files of the news page are ready. Now you need to register the page template in Kentico CMS.

1. Go to **Site Manager -> Development -> Page templates**.
2. Select the **My website** category and click  **New template**.
3. Type *My news template* into the **Template display name** field and click  **Save**.
4. Set the following values on the **General** tab:
  - **Template type**: ASPX page
  - **File name**: ~/CMSTemplates/MySite/NewsPage.aspx
5. Click  **Save**.
6. Switch to the **Sites** tab and assign the page template to **My website**.

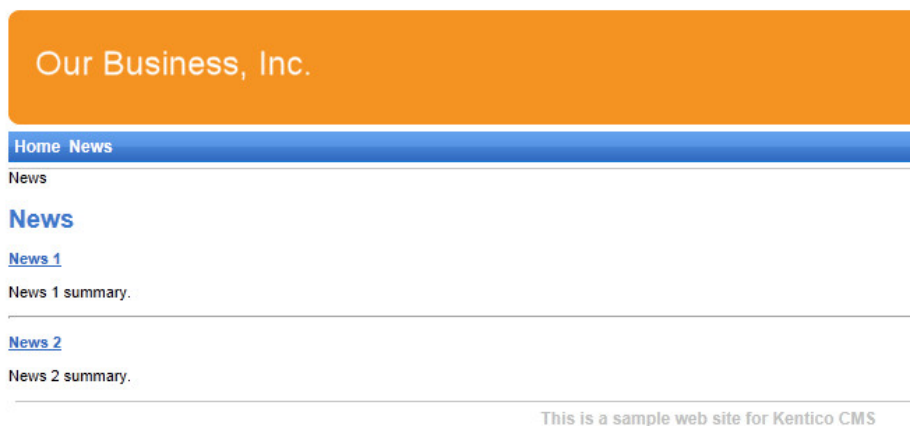
## Adding the news section

1. Go to **CMS Desk -> Content**. Select the root of the content tree (*My website*) and click **New**.
2. Choose the **Page (menu item)** document type.
3. Type in *News* as the **Page name** and choose the **Use existing page template** option. Select the **My website** category and the **My news template** page template.

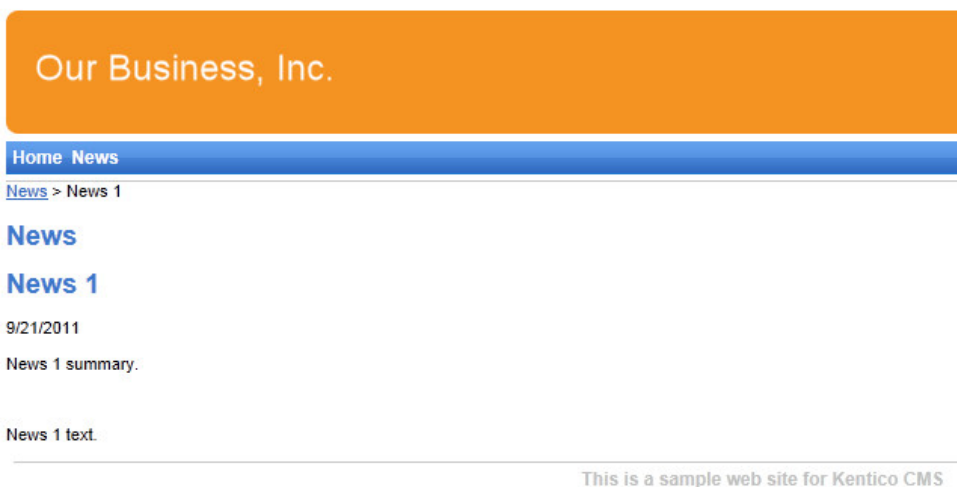


4. Click  **Save** to create the page.
5. Select the **News** page in the content tree, click **New** and choose the **News** document type.
6. Fill in the news document fields with the following values:
  - **News title:** News 1
  - **Release date:** click Now
  - **News summary:** News 1 summary.
  - **News text:** News 1 text.
  - **Publish from/to:** leave the fields blank.
7. Click  **Save and create another** and enter the following values:
  - **News title:** News 2
  - **Release date:** click Now
  - **News summary:** News 2 summary.
  - **News text:** News 2 text.
  - **Publish from/to:** leave the fields blank.
8. Click  **Save**.

If you view the /News page in **Live site** mode, you can see a list of all news documents placed under the **News** section.



This is an example of how content is logically structured in Kentico CMS. If you select a specific news item, the page displays the detail view.



The breadcrumbs at the top of the page show the current path on the website: **News > News 1**. The position is also reflected in the default page URLs:

- The URL of the News page is `~/news.aspx`
- The URL of the News 1 page is `~/news/news-1.aspx`

This makes the website more accessible to both people and search engines, such as Google.

## How it works

1. A visitor arrives on the **/News** page.
2. The **CMSRepeater** control placed on the page template checks if a news document is currently selected (based on the value of the **ClassNames** property).
3. The control finds out that the current document is a *page (menu item)*, so it looks for all underlying news documents and displays them as a list using the **cms.news.preview** transformation.
4. When the visitor selects a particular news item, such as **/News/News 1**, the repeater control uses the **cms.news.default** transformation instead to display the details.



### Path expressions

Listing controls and web parts (such as the *CMSRepeater*) have the **Path** property that specifies which content the component loads and displays. This property supports the following expressions that allow you to select content dynamically:

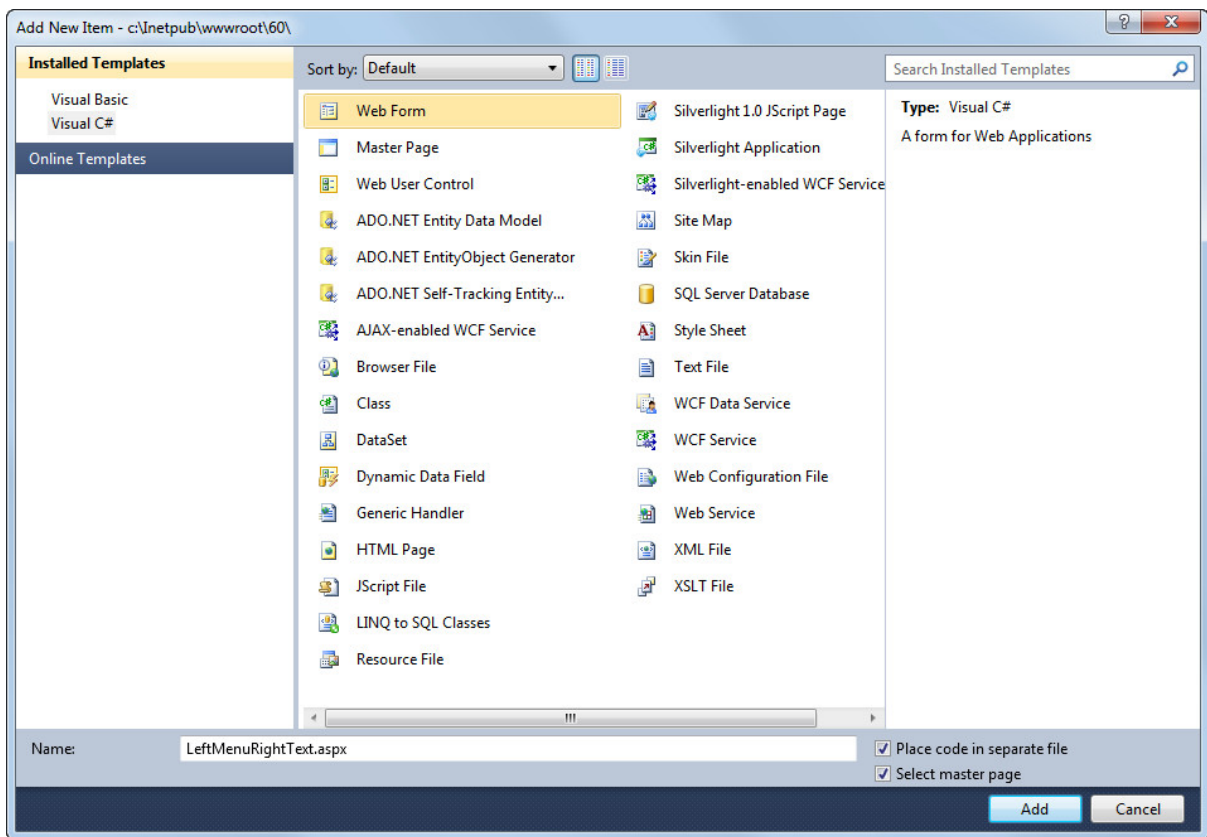
/%	All documents on the website.
/news/%	All documents under <b>/News</b> .
/news/news1	The <b>News1</b> document.
./%	All items under the current document.
./logo	The <b>Logo</b> document under the current document.
./images/%	All documents under the <b>Images</b> document, which is a child of the current document.
../contacts/%	All documents under the <b>Contacts</b> document on the same content level as the current document.
/{0}/%	<p>All documents under the document located on the first level of the current path.</p> <p><u>Example:</u></p> <p>If the currently selected document is:</p> <p><i>/news/news1</i></p> <p>the system evaluates the expression as:</p> <p><i>/news/%</i></p>

## 7.9 Services page

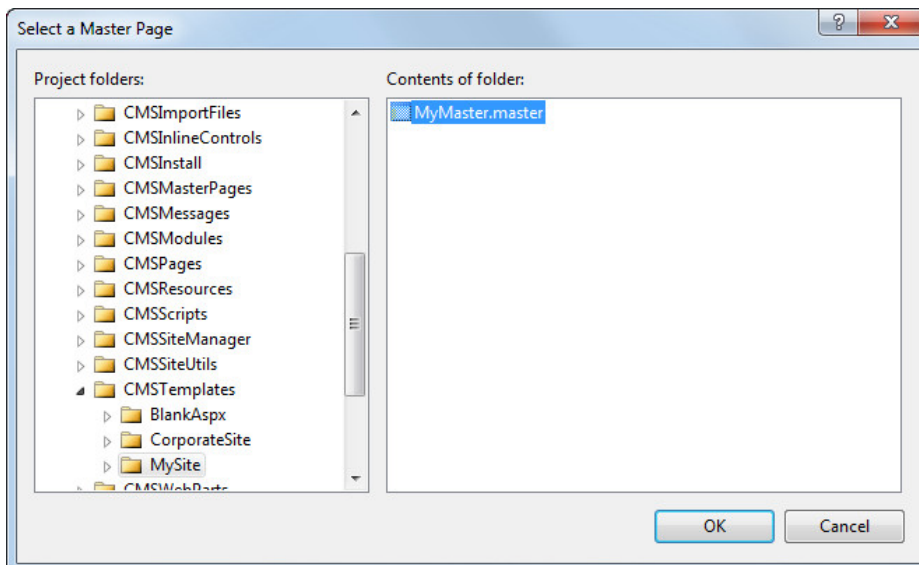
Now we will create a website section for services. The page template used for this section will contain a tree menu on the left and a single editable region.

### Preparing the ASPX source file

1. Edit your web project in Visual Studio, right-click the **CMSTemplates/MySite** folder in the **Solution Explorer** and click **Add new item**.
2. Create a **Web Form** named *LeftMenuRightText.aspx* and check the **Select master page** box.



3. Click **Add** and choose the **MyMaster.master** page from the **CMSTemplates/MySite** folder.

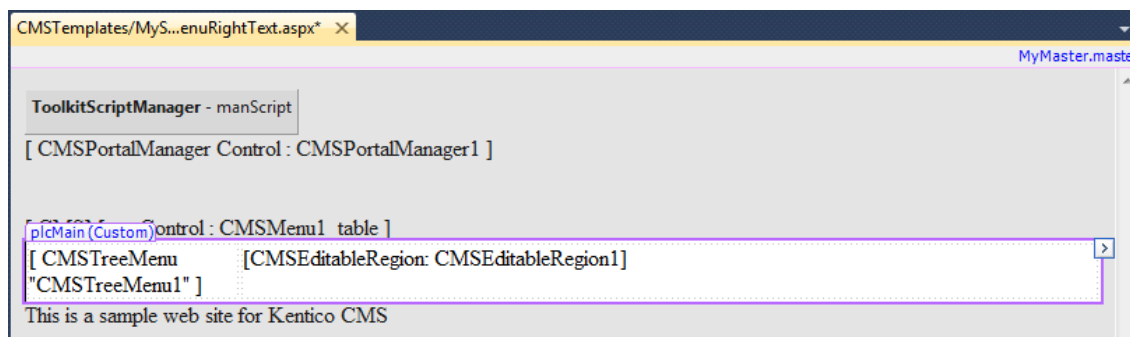


4. Enter the following HTML layout code into the **<asp:Content>** element on the page:

```
<table width="100%">
  <tr valign="top">
```

```
  |
```

5. Switch to the **Design** view where you can see a preview of the page, including the inherited master page. Drag the **CMSTreeMenu** control into the left column and the **CMSEditableRegion** control into the right column.



6. Set the following properties for the controls:

#### CMSTreeMenu:

Property	Value	Description
Path	//{0}/%	Configures the tree menu to display documents starting from the second level of the currently selected path.
MenuItemImageUrl	~/app_themes/mysite/images/bullet.gif	Sets the path of the image displayed next to items in the tree menu.  The ~ character represents the root of the website. This relative path ensures that the control displays the image correctly even if the website's virtual directory name changes.
MenuItemOpenImageURL	~/app_themes/mysite/images/bullet.gif	Specifies the image displayed next to items in the tree menu that belong on the path of the currently selected document.

#### CMSEditableRegion:

Property	Value	Description
RegionType	HTMLEditor	Determines which type of editing interface the control provides. With this option, the editable region works as a WYSIWYG HTML editor.

DialogHeight	400	Sets the height of the editable region in pixels.
RegionTitle	Main Text	Specifies the title displayed in the header of the editable region on the <b>Page</b> tab of CMS Desk.

7. Switch to the code behind of the services page (*LeftMenuRightText.aspx.cs*) and add a reference to the **CMS.UIControls** namespace:

**[C#]**

```
using CMS.UIControls;
```

**[VB.NET]**

```
Imports CMS.UIControls
```

8. Change the class definition so that it inherits from the **TemplatePage** class:

**[C#]**

```
public partial class CMSTemplates_MySite_LeftMenuRightText : TemplatePage
```


**[VB.NET]**

```
Partial Class CMSTemplates_MySite_LeftMenuRightText  
    Inherits TemplatePage
```

9. Save the files.

## Registering the page template


The source files of the services page are ready. Now you need to register the page template in Kentico CMS.

1. Go to **Site Manager -> Development -> Page templates**.
2. Select the **My website** category and click  **New template**.
3. Type *Left menu with right text* into the **Template display name** field and click  **Save**.
4. Set the following values on the **General** tab:
  - **Template type**: ASPX page
  - **File name**: ~/CMSTemplates/MySite/LeftMenuRightText.aspx
5. Click  **Save**.

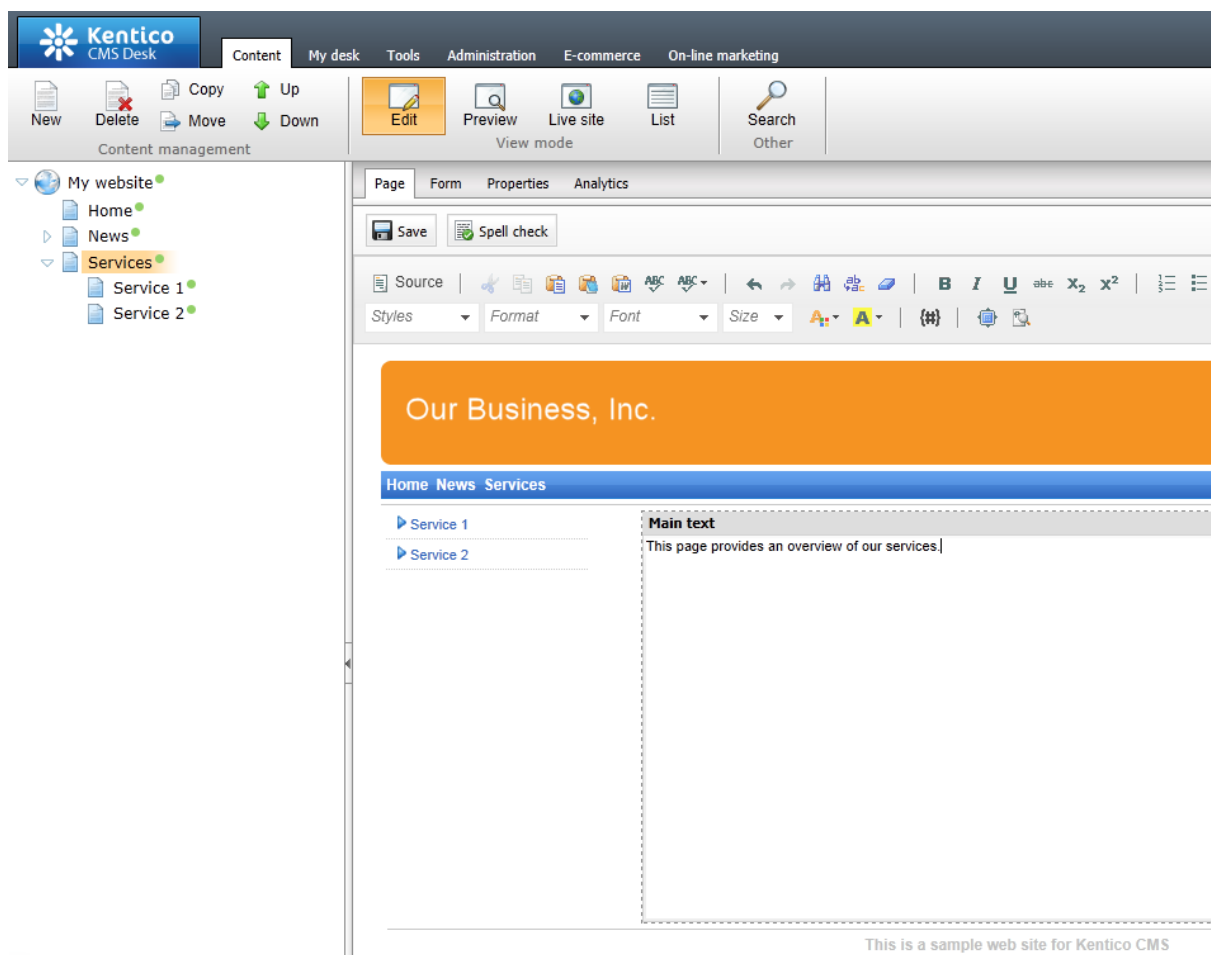
6. Switch to the **Sites** tab and assign the page template to **My website**.

## Adding the services section

Now that the page template is ready, you can start adding pages based on it.

1. Go to **CMS Desk -> Content**. Select the root of the content tree (*My website*) and click **New**.
2. Choose the **Page (menu item)** document type.
3. Type in *Services* as the **Page name** and choose the **Use existing page template** option. Select the **My website** category and the **Left menu with right text** page template.
4. Click  **Save** to create the page.


The **Page** tab of the Services page opens, where you can enter text content into the editable region on the right.



## Creating sub-pages



You can use the same page template to add separate pages containing information about individual services under the **Services** page.

1. Select the **/Services** page in the content tree and click **New** on the main toolbar.
2. Choose the **Page (menu item)** document type.
3. Type in *Service 1* as the **Page name** and choose the **Use existing page template** option. Select the **My website** category and the **Left menu with right text** page template.
4. Click  **Save and create another** and repeat the same process to add any number of pages dedicated to particular services.

All of the service pages use the same template as the main overview page (*/Services*). The page template provides the tree menu for navigation in the services section and an editable region. The system stores the text of the region separately for every document, so you can enter unique content on each page.

## 7.10 Products page


### 7.10.1 Overview

Now we will add a products section displaying a list of computers and their technical specification. You will learn how to:

1. Create a [new document type](#) representing computer products
2. Write [transformations](#)
3. [Create a page](#) displaying a list of computer products

### 7.10.2 New document type

Each document in the Kentico CMS repository is of a certain type, such as news, product, article, etc. Every document type has its own data fields. Our document type describing computer products will have fields storing the computer name, processor type, RAM size, disk size and product image.

1. Go to **Site Manager -> Development -> Document types** and click  **New document type**. This starts the **New document type** wizard.
2. Enter the following values in **Step 1**:
  - **Document type display name**: *Computer* (the system displays this name to users in the administration interface)
  - **Document type code name**: *custom.computer* (*custom* is a namespace to distinguish your document types from the default system types that use the *cms* namespace, *computer* is the identifier of the document type. You will use this value in the properties of controls later.)

The screenshot shows the 'New document type' wizard in the Kentico CMS 7.0 Site Manager. The interface includes a top navigation bar with 'Development' selected, and a left sidebar with various development tools. The main content area is titled 'Step 1 | General' and contains the following fields:

- Document type display name:
- Document type code name:  namespace  document type

A 'Next >' button is located at the bottom right of the form.

Click **Next**.


3. In **Step 2**, you need to specify the name of the database table where the system will store the data of computer documents. You also need to enter the name of the table's primary key field. Leave the default values.

The screenshot shows the 'New document type' wizard in the Kentico CMS 7.0 Site Manager, Step 2: Data type. The interface includes a top navigation bar with 'Development' selected, and a left sidebar with various development tools. The main content area is titled 'Step 2 | Data type' and contains the following fields:

- The document type has custom fields
  - Table name:
  - Primary key name:
  - Inherits fields from document type:
- The document type is only a container without custom fields

A 'Next >' button is located at the bottom right of the form.

Click **Next**. The wizard creates a new database table for computer documents.

4. In **Step 3**, you need to define the fields of the document type (columns of the table). Click **New attribute** (+) to create the following fields. For each field, enter the values, click  **Save** and repeat the procedure until you have all the listed fields defined.

- **Column name:** ComputerName
- **Attribute type:** Text
- **Attribute size:** 200
- **Field caption:** Computer name
- **Form control:** Text box
  
- **Column name:** ComputerProcessorType
- **Attribute type:** Text
- **Attribute size:** 200
- **Field caption:** Processor type
- **Form control:** Drop-down list
- **Editing control settings -> Data source:** select **Options** and enter the following items into the text area, one per line:  
Athlon;Athlon  
Pentium XEON;Pentium XEON  
Pentium Core 2 Duo;Pentium Core 2 Duo
  
- **Column name:** ComputerRamSize
- **Attribute type:** Integer number
- **Field caption:** RAM (MB)
- **Form control:** Text box
  
- **Column name:** ComputerHddSize
- **Attribute type:** Integer number
- **Field caption:** HDD (GB)
- **Form control:** Text box
  
- **Column name:** ComputerImage
- **Attribute type:** File
- **Allow empty value:** yes (checked)
- **Field caption:** Image
- **Form control:** Upload file

**Step 3** Fields  
 Please define custom attributes of the document type and their appearance in the editing form. You can define attributes, such as product number, product weight, press release text, etc.

ComputerID\*  
ComputerName\*  
ComputerProcessorType\*  
ComputerRamSize\*  
ComputerHddSize\*  
**ComputerImage**

**Quick links:**  
[Database](#)  
[Field appearance](#)  
[Editing control settings](#)  
[CSS styles](#)

The changes were saved.

**Database**

Column name:

Attribute type:

Attribute size:

Allow empty value:

Display attribute in the editing form

**Field appearance**

Field caption:

Form control:

Field description:

Click **Next**.

5. In **Step 4**, choose the *ComputerName* field as the **Document name source**.

This means that when a user creates a new computer document, the system automatically fills in the document name based on the ComputerName value. The document name appears in site navigation and in the CMS Desk content tree.

**Step 4** Additional settings  
 Please choose the source field that will be used as a document name. You can choose either one of the custom fields or you can choose to use document name as a separate field.

Document name source:

Click **Next**.

6. In **Step 5**, select the document types that will be supported as parents for computer documents in the

content tree. Click **Add document types**, select the **Page (menu item)** document type and click **OK**.

This means that users are only allowed to place computer documents under pages, not under articles, news items or other document types.

### Step 5

#### Parent types

Please select document types under which this document template can be placed.

<input type="checkbox"/>	Document type name
<input type="checkbox"/>	Page (menu item) (CMS.Menuitem)

[Remove selected](#) [Add document types](#) ▼

[Next >](#)

Click **Next**.

7. In **Step 6**, assign the document type to all websites where you wish to use it. Click **Add sites**, choose **My website** in the selection dialog and click **OK**.

### Step 6

#### Sites

Please select sites where this document type can be used.

<input type="checkbox"/>	Site name
<input type="checkbox"/>	My website

[Remove selected](#) [Add sites](#) ▼

[Next >](#)

Click **Next**.

8. **Step 7** of the wizard allows you to specify how the Smart search module indexes documents of this type and displays them in search results. Select the following values in the drop-downs:

- **Title field:** ComputerName
- **Content field:** DocumentContent
- **Image field:** ComputerImage
- **Date field:** DocumentCreatedWhen

**Step 7** | **Search options**  
Please set search fields for Smart search module.

Title field:  ▼  
 Content field:  ▼  
 Image field:  ▼  
 Date field:  ▼

Set automatically

Field name	Content	Searchable	Tokenized	Custom search name
ComputerID	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="text"/>
ComputerName	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="text"/>
ComputerProcessorType	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="text"/>
ComputerRamSize	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="text"/>
ComputerHddSize	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="text"/>

**Next >**

Leave the default values for the rest of the configuration options and click **Next**.

9. Click **Finish** to complete the creation of the new document type.

**Step 8** | **The wizard has finished**

The setup has finished the following steps:

- › The new document type was created.
- › The new editing form was created.
- › The document types were added among allowed child types of the new document type.
- › The sites were selected where this document type can be used.
- › The default queries were created.
- › The default ASCX transformations were created.
- › The default permission names were created.
- › The default icon was created.
- › Document smart search specification was created.

**Finish**

The wizard automatically creates not only the database table, but also the SQL queries for SELECT,

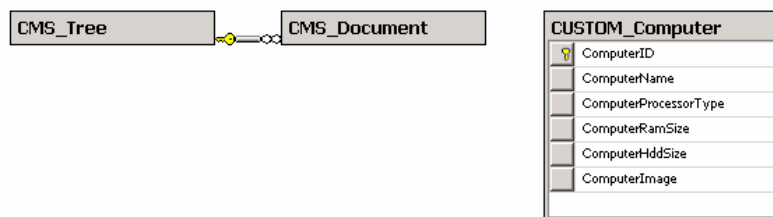
INSERT, UPDATE, DELETE operations and a several default transformation.



### How does the system store document content?

The CMS stores document content and all related data in three database tables:

- **CMS\_TREE** (content tree structure)
- **CMS\_Document** (general document properties, metadata and editable region content)
- The dedicated document type table - in this case **CUSTOM\_Computer** (stores the values of the document type's specific fields)



The system automatically ensures all operations related to these tables. **You can easily write standard SQL SELECT queries to retrieve data** from the repository (i.e. the Microsoft SQL Server database).

You have learned how to create new document types. Continue in the [Transformations](#) topic, which describes how to define the format used to display document data on the website.

### 7.10.3 Transformations

Now that you have created the new document type, you need to prepare the transformations that page components will use to display computer products on the website.

1. Go to **Site Manager -> Development -> Document types** and edit (✎) the **Computer (custom.computer)** document type.
2. Switch to the **Transformations** tab.

Document type properties

> Document types > Computer

General Fields Form Transformations Queries Child types Sites E-commerce Alternative forms Search fields Documents Versions

New transformation New hierarchical transformation

Actions	Transformation name	Transformation type
	AtomItem	ASCX
	Default	ASCX
	Preview	ASCX
	RSSItem	ASCX

The New document type wizard has created several default transformations, which you can use as a base for your own transformations.

3. **Edit** () the **Default** transformation, clear the original code and replace it with the following:

```
<h1>
    <%# Eval("ComputerName") %>
</h1>
<table>
    <tr>
        <td>
            Processor:
        </td>
        <td>
            <%# Eval("ComputerProcessorType") %>
        </td>
    </tr>
    <tr>
        <td>
            RAM (MB):
        </td>
        <td>
            <%# Eval("ComputerRamSize") %>
        </td>
    </tr>
    <tr>
        <td>
            HDD (GB):
        </td>
        <td>
            <%# Eval("ComputerHddSize") %>
        </td>
    </tr>
    <tr>
        <td>
            Image:
        </td>
        <td>
            <%# GetImage("ComputerImage") %>
        </td>
    </tr>
</table>
```



```
</tr>
</table>
```

ASCX transformation code is similar to standard *ItemTemplate* elements that you may already be familiar with from using ASP.NET Repeater or DataList controls. It combines HTML with ASP.NET commands and data binding expressions (*Eval*). You may also use built-in methods that simplify various tasks, such as **GetImage**. For more information about the available transformation methods, click the [? Transformation examples](#) link above the code editor.

You will use the **Default** transformation for displaying the *details* of individual computer products.

4. Click  **Save**.

5. Return to the transformation list and edit the **Preview** transformation. Clear the default code and add the following code instead:

```
<div style="text-align:center;padding: 8px;margin: 4px;border: 1px solid #CCCCCC">
  <h2>
    <a href="<%# GetDocumentUrl() %>"><%# Eval("ComputerName") %></a>
  </h2>
  <%# GetImage("ComputerImage", 120) %>
</div>
```

Note the code used to create the link to specific documents. It consists of a standard HTML link tag and inserts the appropriate URL and link text dynamically:

```
<a href="<%# GetDocumentUrl() %>"><%# Eval("ComputerName") %></a>
```

You can generate an image tag containing the file uploaded into the given document's **ComputerImage** field using the **GetImage** method. The sample code calls the method with a parameter that ensures automatic server-side resizing of the image's longest side to 120 pixels:

```
<%# GetImage("ComputerImage", 120) %>
```

You will use the **Preview** transformation for displaying the *list* of computer documents on the main products page.



### Entering field names in transformations

When writing ASCX transformations, you often need to specify the names of data fields as parameters of the *Eval* data binding expression or other methods, such as *ComputerName* and *ComputerImage* in the examples above.

You can either type the names manually, or press the **CTRL + SPACE** key combination to access a list of available document fields and related objects.

6. Click  **Save**.

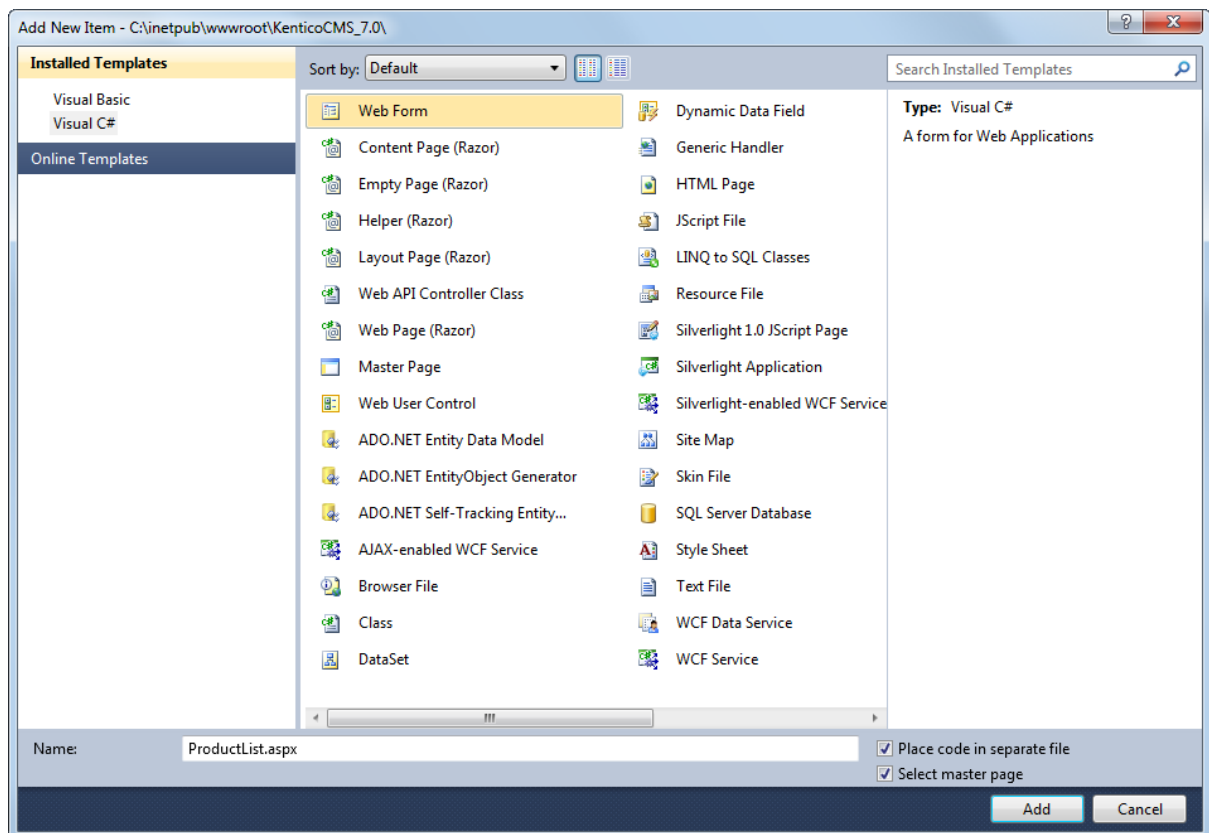
You have learned how to write transformations for displaying the content of structured documents. Continue in the [Page template](#) topic to create a website section that uses the *Preview* and *Default* transformations.

## 7.10.4 Page template

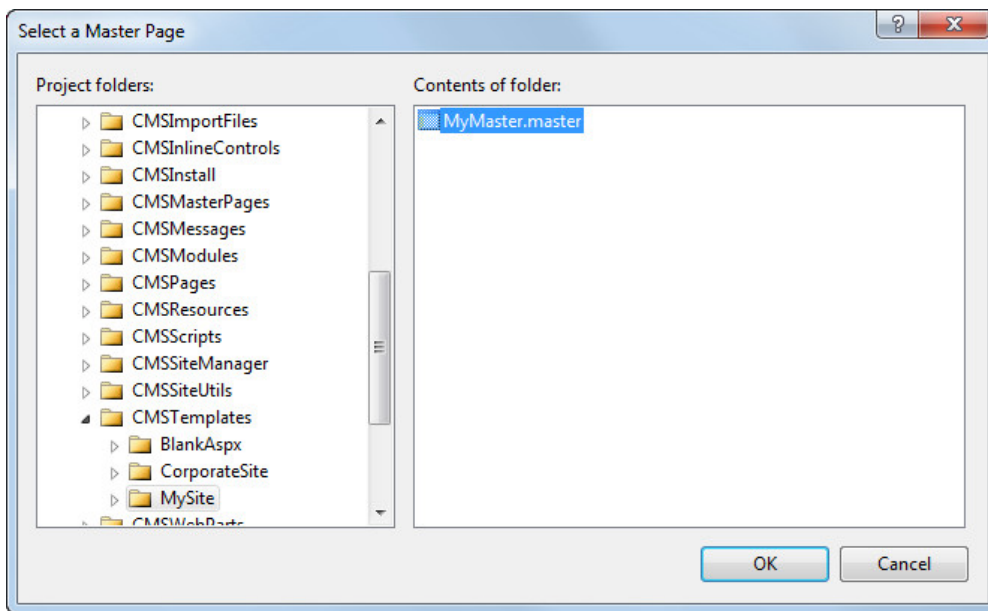
This topic describes how to create the product list page and publish computer specifications on the website.

### Preparing the ASPX source file for the products template

1. Edit your web project in Visual Studio, right-click the **CMSTemplates/MySite** folder in the **Solution Explorer** and click **Add new item**.
2. Create a **Web Form** named *ProductList.aspx* and check the **Select master page** box.



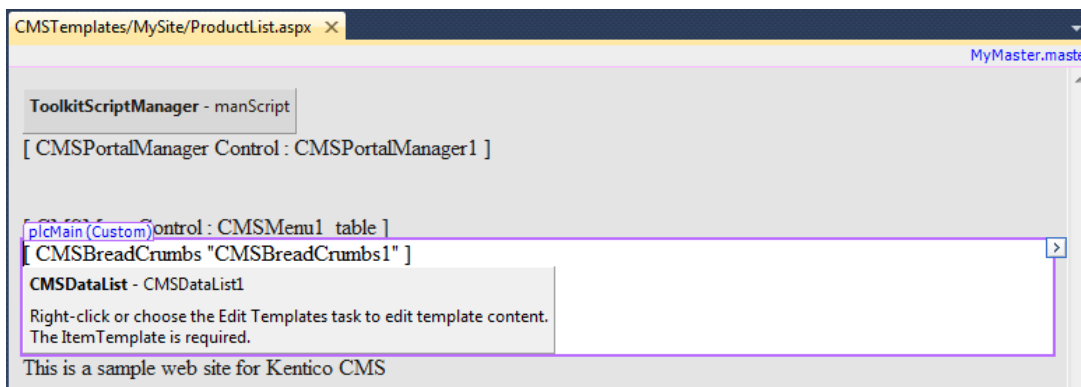
3. Click **Add** and choose the **MyMaster.master** page from the **CMSTemplates/MySite** folder.



4. Drag the following controls inside the `<asp:Content>` element of the product list page:

- **CMSBreadCrumbs**
- **CMSDataList**

If you view the web form on the **Design** tab, you should see the structure of the page as shown below:



5. Set the properties of the **CMSDataList** control according to the table below:

Property	Value	Description
ClassNames	custom.computer	Configures the datalist to display only documents of the <i>custom.computer</i> type (created in the <a href="#">New document type</a> topic).
OrderBy	ComputerName ASC	Sets the SQL ORDER BY clause that the control uses when loading data. As a result, the control displays items in ascending alphabetical order based on the <i>ComputerName</i> values.
TransformationName	custom.computer.p	Assigns the transformation that the datalist uses to

	review	display the list of computer products (as defined in <a href="#">Transformations</a> ).
SelectedItemTransformationName	custom.computer.default	When a user selects a specific computer document on the website, the control displays the details according to the specified transformation.
RepeatColumns	2	Configures the datalist to display 2 items per row in list mode.

6. Switch to the code behind of the product list page (*ProductList.aspx.cs*) and add a reference to the **CMS.UIControls** namespace:

[C#]

```
using CMS.UIControls;
```

[VB.NET]

```
Imports CMS.UIControls
```

7. Change the class definition so that it inherits from the **TemplatePage** class:

[C#]

```
public partial class CMSTemplates_MySite_ProductList : TemplatePage
```

[VB.NET]

```
Partial Class CMSTemplates_MySite_ProductList
    Inherits TemplatePage
```

8. Save the files.

## Registering the page template

The source files of the product list page are ready. Now you need to register the page template in Kentico CMS.

1. Go to **Site Manager -> Development -> Page templates**.
2. Select the **My website** category and click  **New template**.
3. Type *Product list* into the **Template display name** field and click  **Save**.
4. Set the following values on the **General** tab:

- **Template type:** ASPX page
- **File name:** ~/CMSTemplates/MySite/ProductList.aspx

5. Click  **Save**.


6. Switch to the **Sites** tab and assign the page template to **My website**.

## Adding the products section

1. Go to **CMS Desk -> Content**. Select the root of the content tree (*My website*) and click **New**.

2. Choose the **Page (menu item)** document type.

3. Type in *Products* as the **Page name** and choose the **Use existing page template** option. Select the **My website** category and the **Product list** page template.

4. Click  **Save** to create the page.

5. Select the **Products** page in the content tree, click **New** and choose the **Computer** document type.

6. Fill in the computer document fields with the following values:

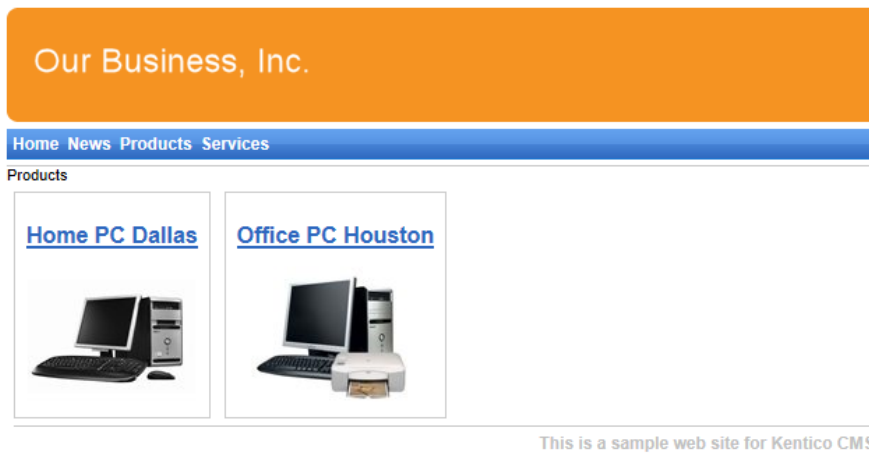
- **Computer name:** Home PC Dallas
- **Processor type:** Athlon
- **RAM (MB):** 2048
- **HDD (GB):** 160
- **Image:** upload an image (you can find sample images in the <Kentico CMS installation>\CodeSamples\SampleWebTemplate\Computer\_Images folder)
- **Publish from/to:** leave the values blank

7. Click  **Save and create another** and enter the following values:

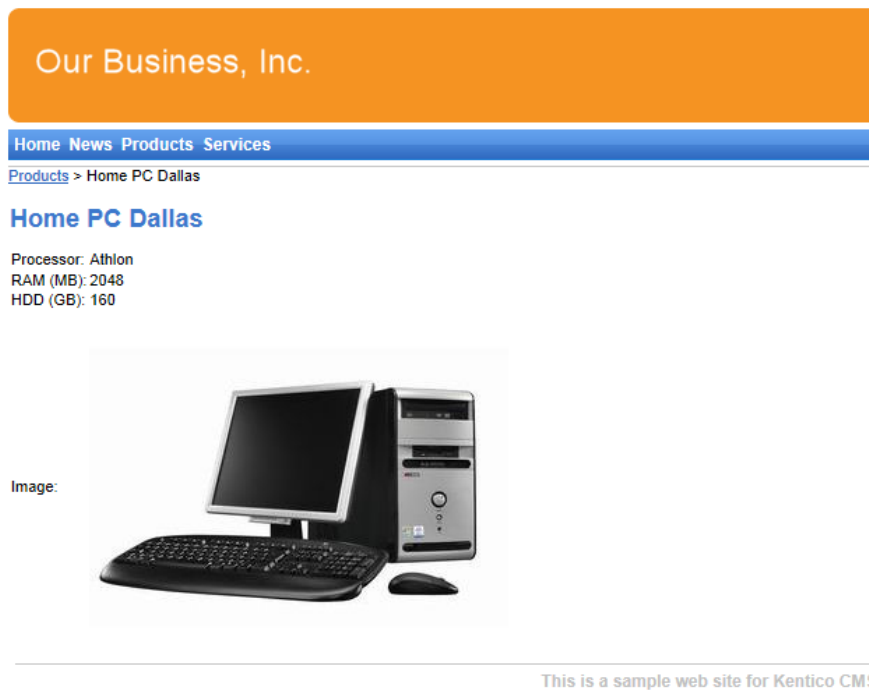
- **Computer name:** Office PC Houston
- **Processor type:** Pentium Core 2 Duo
- **RAM (MB):** 4096
- **HDD (GB):** 200
- **Image:** upload an image (you can find sample images in the <Kentico CMS installation>\CodeSamples\SampleWebTemplate\Computer\_Images folder)
- **Publish from/to:** leave the values blank

8. Click  **Save**.

If you view the */Products* page in **Live site** mode, you can see a list of the two computer products (formatted according to the **custom.computer.preview** transformation):



When you click the title of a specific computer, the page displays the detail view (using the `custom.computer.default` transformation):








## 7.11 Search page

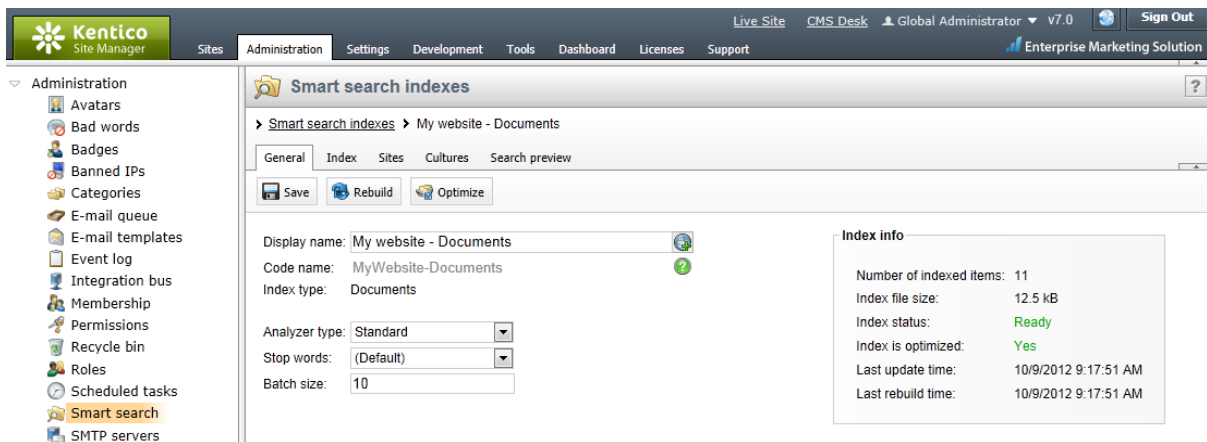
Kentico CMS allows users to perform index-based searches through all document content in the Kentico CMS repository, as well as other types of data. This topic describes how to add a basic search page to your website.

### Creating a smart search index

Before you can use the search, you need to add a smart search index covering the website's documents.

1. Go to **Site Manager** -> **Administration** -> **Smart search**.
2. Click  **New index**.
3. Fill in the following details for the search index:
  - **Display name:** My website - Documents
  - **Index type:** Documents
  - **Analyzer type:** Standard
  - **Stop words:** Default
4. Click  **Save**. The index's editing interface opens.
5. Open the **Index** tab and click  **Add allowed content**.
6. Type `/%` into the **Path** field and click  **Save**. This ensures that the index includes all documents on the website.
7. Switch to the **Sites** tab and assign the index to **My website**.
8. Switch to the **Cultures** tab and choose the default culture of your site (typically *English - United States*).
9. Open the **General** tab and click  **Rebuild**.

Once the system rebuilds the index, you can start using it on the website. The **Index info** box on the right side of the tab displays the current status of the index and other relevant information.



The screenshot shows the Kentico Site Manager Administration interface. The top navigation bar includes 'Sites', 'Administration', 'Settings', 'Development', 'Tools', 'Dashboard', 'Licenses', and 'Support'. The left sidebar lists various administration tasks, with 'Smart search' highlighted. The main content area is titled 'Smart search indexes' and shows the configuration for 'My website - Documents'. The configuration includes fields for Display name, Code name, Index type, Analyzer type, Stop words, and Batch size. The Index info box on the right shows the number of indexed items (11), index file size (12.5 kB), index status (Ready), and last update/rebuild times.



### Searching through uploaded text files

You can also configure the CMS to search the text inside uploaded files, such as PDF, DOC or XLS documents. For additional details, see the following chapter in the Kentico CMS Developer's Guide: [Installation and deployment -> Additional configuration tasks -> Configuration of full-text search in files](#)

You do not need to configure this option now, since we will only use the basic

document search in this tutorial.

## Preparing the ASPX source files for the search page

1. Edit your web project in Visual Studio, right-click the **CMSTemplates/MySite** folder in the **Solution Explorer** and click **Add new item**.
2. Create a **Web Form** named *SearchPage.aspx* and check the **Select master page** box.
3. Click **Add** and choose the **MyMaster.master** page from the **CMSTemplates/MySite** folder.
4. Add the following directive to the beginning of the page code:

```
<%@ Register src="~/CMSWebParts/SmartSearch/SearchDialogWithResults.ascx"
tagname="SearchDialogWithResults" tagprefix="cms" %>
```

This registers the *Smart search dialog with results* web part as a user control for use on the ASPX template.

5. Copy the following code inside the **<asp:content>** element of the page:

```
<h1>Search</h1>

<cms:SearchDialogWithResults ID="SearchDialogWithResults1" runat="server"
TransformationName="cms.root.smartsearchresultswithimages" Indexes="MyWebsite-
Documents" />
```

This adds a heading and the user control (web part) that provides search functionality and displays the results. The control uses the *My website - Documents* search index created in the previous section, which is assigned through the **Indexes** property (identified by the index code name).

6. Switch to the code behind of the search page (*SearchPage.aspx.cs*) and add a reference to the **CMS.UIControls** namespace:

**[C#]**

```
using CMS.UIControls;
```

**[VB.NET]**

```
Imports CMS.UIControls
```

7. Change the class definition so that it inherits from the **TemplatePage** class:



**[C#]**

```
public partial class CMSTemplates_MySite_SearchPage : TemplatePage
```




**[VB.NET]**

```
Partial Class CMSTemplates_MySite_SearchPage  
    Inherits TemplatePage
```



8. Save the search page files.

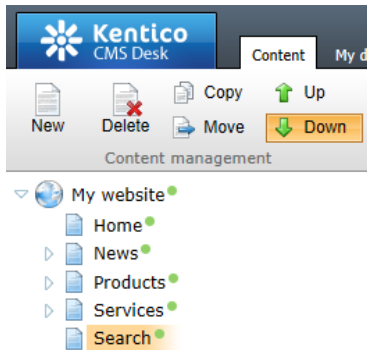
## Registering the page template

The source files of the search page are ready. Now you need to register the page template in Kentico CMS.

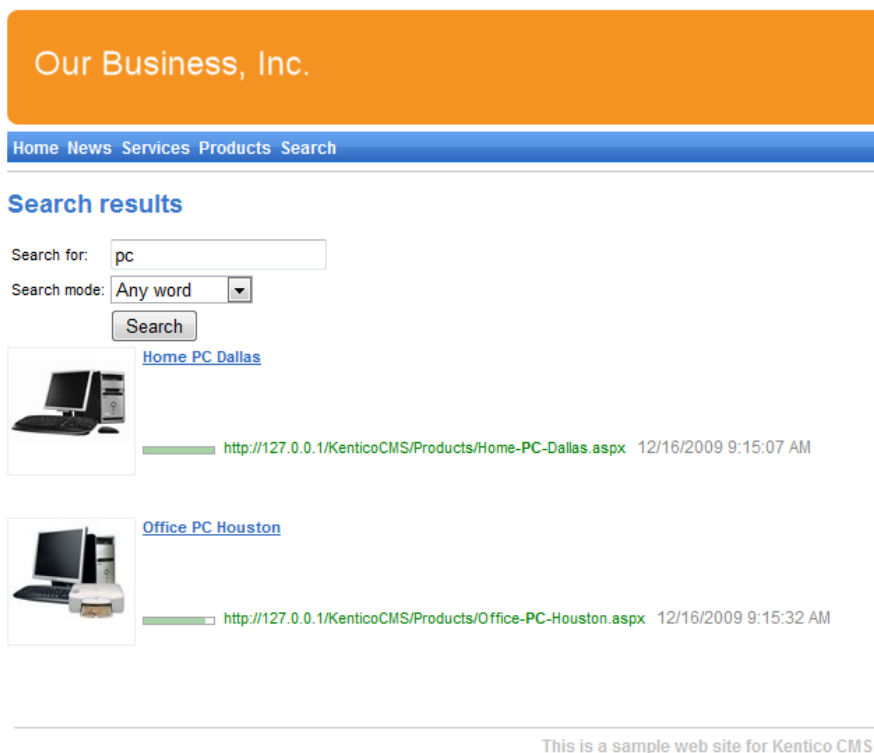
1. Go to **Site Manager -> Development -> Page templates**.
2. Select the **My website** category and click  **New template**.
3. Type *Search page* into the **Template display name** field and click  **Save**.
4. Set the following values on the **General** tab:
  - **Template type**: ASPX page
  - **File name**: ~/CMSTemplates/MySite/SearchPage.aspx
5. Click  **Save**.
6. Switch to the **Sites** tab and assign the page template to **My website**.

## Adding the search page

1. Go to **CMS Desk -> Content**. Select the root of the content tree (*My website*) and click **New**.
2. Choose the **Page (menu item)** document type.
3. Type in *Search* as the **Page name** and choose the **Use existing page template** option. Select the **My website** category and the **Search page** template.
4. Click  **Save** to create the page.
5. Click the **Down** () arrow on the main toolbar to move the **Search** page to the end of the document list.



To try out the search functionality, view the `/Search` page in **Live site** mode. Type *PC* into the **Search for** box and click **Search**.



When you click a search result, the system redirects you to the corresponding page.



### Modifying the search results format



If you prefer a different design of the search results, you can modify the format by editing the **SmartSearchResults** (or **SmartSearchResultsWithImages**) transformation in **Site Manager -> Development -> Document types -> Root -> Transformations**.

## 7.12 Secured section for partners

Kentico CMS provides a way to create secured site sections that can only be viewed by users who have a valid user name and password. This topic describes how to create a logon page for the purposes of user authentication and registration, as well as a secured page accessible only by logged in users.

### Adding the secured partners page

Start by adding a new secured page that requires authentication:

1. Go to **CMS Desk -> Content**. Select the root of the content tree (*My website*) and click **New**.
2. Choose the **Page (menu item)** document type.
3. Type in *Partners* as the **Page name** and choose the **Use existing page template** option. Select the **My website** category and the **Left menu with right text** page template. This page reuses the template originally created for the website's [Services page](#).
4. Click  **Save** to create the page.
5. Select the **Page** tab and type the following text into the editable region: *This is a secured page for partners*.
6. Click  **Save**.
7. Open the **Properties -> Security** tab of the *Partners* document.
8. Select **Yes** for the **Requires authentication** property in the **Access** section at the bottom of the dialog and click **OK**.

This ensures that only authenticated (logged in) users can access the page.

### Creating the logon page

Now we will build a page where users can sign in to the website and anonymous visitors can register as new users.

### Preparing the source files

1. Edit your web project in Visual Studio, right-click the **CMSTemplates/MySite** folder in the **Solution Explorer** and click **Add new item**.
2. Create a **Web Form** named *LogonPage.aspx* and check the **Select master page** box.
3. Click **Add** and choose the **MyMaster.master** page from the **CMSTemplates/MySite** folder.
4. Enter the following HTML layout code into the **<asp:Content>** element on the page:

```
<table border="0" width="100%">
  <tr valign="top">
```

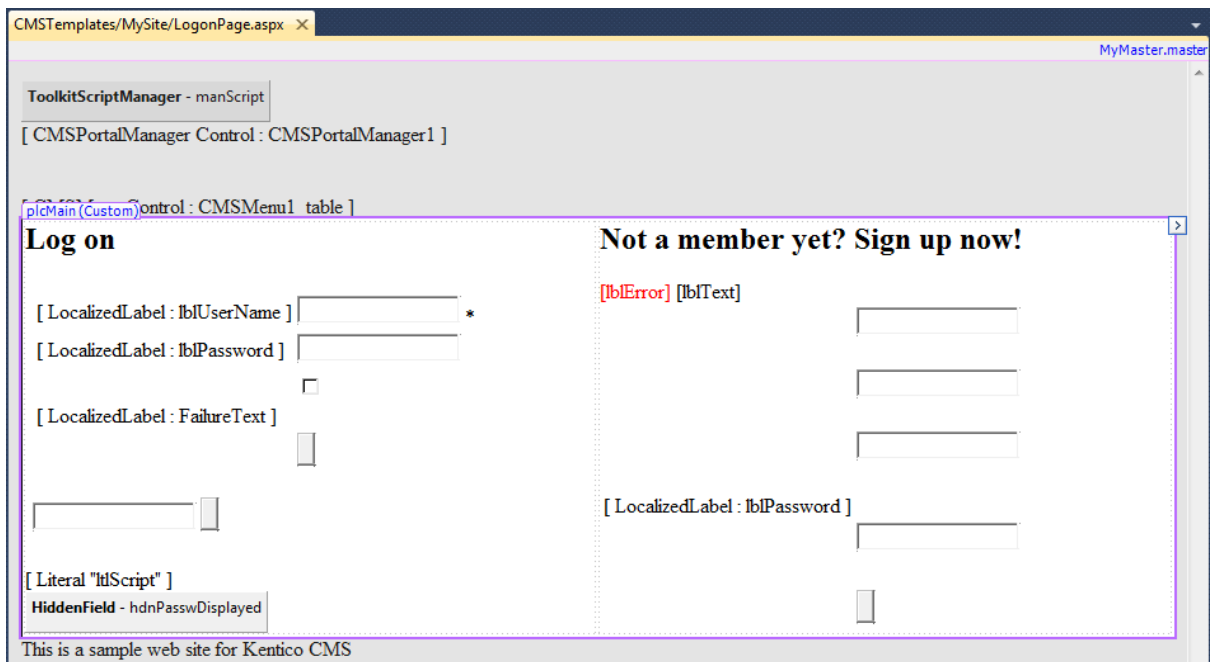
```

<td style="width:50%">
    <h2>Log on</h2>
</td>
<td style="width:50%">
    <h2>Not a member yet? Sign up now!</h2>
</td>
</tr>
</table>

```

5. Switch to the **Design** view and drag the following web parts (user controls) from the **Solution Explorer** into the left and right columns respectively:

- ~/CMSWebParts/Membership/Logon/LogonForm.ascx
- ~/CMSWebParts/Membership/Registration/RegistrationForm.ascx



6. Set the following properties for the controls:

#### LogonForm:

Property	Value	Description
AllowPasswordRetrieval	true	Configures the logon form to display a link that allows users to recover forgotten passwords or generate new ones via e-mail.
SendEmailFrom	no-reply@localhost.local	Sets the sender address for the password recovery e-mails.

#### RegistrationForm:

Property	Value	Description
EnableUserAfterRegistration	true	Configures the control to automatically enable new user accounts after registration.

7. Switch to the code behind of the logon page (*LogonPage.aspx.cs*) and add a reference to the **CMS.UIControls** namespace:

**[C#]**

```
using CMS.UIControls;
```

**[VB.NET]**

```
Imports CMS.UIControls
```

8. Change the class definition so that it inherits from the **TemplatePage** class:

**[C#]**




```
public partial class CMSTemplates_MySite_LogonPage : TemplatePage
```

**[VB.NET]**



```
Partial Class CMSTemplates_MySite_LogonPage  
    Inherits TemplatePage
```

9. Save the files.

## Registering the logon page template

1. Go to **Site Manager -> Development -> Page templates**.
2. Select the **My website** category and click  **New template**.
3. Type *Logon page* into the **Template display name** field and click  **Save**.
4. Set the following values on the **General** tab:
  - **Template type**: ASPX page
  - **File name**: ~/CMSTemplates/MySite/LogonPage.aspx
5. Click  **Save**.
6. Switch to the **Sites** tab and assign the page template to **My website**.


## Adding the logon page document

1. Go to **CMS Desk -> Content**. Select the root of the content tree (*My website*) and click **New**.
2. Choose the **Folder** document type.
3. Type in *Special pages* as the **Document name** and click  **Save**.
4. Click **New** again and select the **Page (menu item)** document type.
5. Type in *Logon* as the **Page name** and choose the **Use existing page template** option. Select the **My website** category and the **Logon page** template.
6. Click  **Save** to create the page.

Because you placed the Logon page under a folder, it does not show up in the website's navigation menu. The menu control on the master page is configured to only display documents of the *Page (menu item)* type. You can use folders to store pages that have a specific purpose, but are not part of the website's regular content.

## Setting the website's logon page

When an anonymous visitor attempts to access a secured page that requires authentication (such as the *Partners* page on your sample website), the system redirects them to a logon page. By default, websites use the system page that appears when signing into CMS Desk. However, you can configure each website to use its own custom logon page.

1. Go to **Site Manager -> Settings** and click the **Security & Membership** category in the settings tree.
2. Select **My website** in the **Site** drop-down menu.
3. Uncheck the **Inherit from global settings** box next to the **Website logon page URL** setting and type in `~/Special-pages/Logon.aspx`. This is the relative URL of the logon page that you added to the website.
4. Click  **Save**.

The screenshot shows the Kentico Site Manager interface for the 'Security & Membership' settings. The left sidebar shows a tree view with 'Security & Membership' selected. The main content area is titled 'Security & Membership' and includes a 'Save' button and a 'Reset these settings to default' link. The settings are organized into four sections:

- General:**
  - Administrator's e-mail:   Inherit from global settings
  - Send membership reminder (days):   Inherit from global settings
- Registrations:**
  - Reserved user names:   Inherit from global settings
  - Registration requires e-mail confirmation:   Inherit from global settings
  - Registration requires administrator's approval:   Inherit from global settings
  - Delete non-activated user after (days):   Inherit from global settings
  - Require unique user e-mails:   Inherit from global settings
- On-line users:**
  - Update on-line users (minutes):   Inherit from global settings
- Content:**
  - Check page permissions:   Inherit from global settings
  - Website logon page URL:   Inherit from global settings
  - Access denied page URL:   Inherit from global settings

The website's logon page is now ready.

## Adding a sign out button to the website

The website now allows users to log in, so you should also provide a way to log out. You can do this by adding components to the website's master page.

1. Open your web project in Visual Studio and edit the **MyMaster.master** master page (in the **CMSTemplates/MySite** folder).
2. Switch to the **Design** view and drag the following web parts (user controls) from the **~/CMSWebParts/Membership/Logon/** folder in the **Solution Explorer** and place them before the **CMSSMenu** control (inside the `<div class="MainMenu">` element):

- **SignOutButton.ascx**
- **CurrentUser.ascx**

The screenshot shows the Visual Studio Design view of the `MyMaster.master` master page. The code is as follows:

```

ToolkitScriptManager - manScript
[ CMSPortalManager Control : CMSPortalManager1 ]

[ Literal "litScript" ] [btnSignOut] [btnSignOutLink][lbLabel] [ Literal "litSignLink" ] [ CMSMenu Control : CMSMenu1_table ]

This is a sample web site for Kentico CMS

```

3. Set the following properties for the controls:

**SignOutButton:**

Property	Value	Description
ShowOnlyWhenAuthenticated	true	Ensures that master page only displays the sign out button when the site is being viewed by an authenticated users.
CssClass	Right	Sets the name of the CSS class applied to the button.

**CurrentUser:**

Property	Value	Description
ShowOnlyWhenAuthenticated	true	Ensures that master page only displays the current user information when the site is being viewed by an authenticated users.
CssClass	CurrentUser Right	Sets the names of the CSS classes applied to the label.

4. Save the changes.

5. Go to **Site Manager -> Development -> CSS Stylesheets** and edit (✎) the **My site stylesheet**.

6. Add the following class definitions to the stylesheet:

```
.CurrentUser
{
  color: white;
  padding-top: 4px;
}

.Right
{
  float: right;
  padding-right: 5px;
}
```

7. Click  **Save**.

The **Sign out** button and **CurrentUser** control are now visible for signed in users on all pages on the website.

## Result - Logging in to the website

Now that you have added the logon page, secured section and sign out button to the website, you can test the new functionality from the perspective of a live site user.

1. **Sign out** of Site Manager so you can view the website as a public visitor.

2. Click **Partners** in the main menu. The logon page appears.



The screenshot shows the login and registration interface for 'Our Business, Inc.'. The page has an orange header with the company name and a blue navigation bar with links for Home, News, Partners, Products, Services, and Search. Below the navigation bar, there are two main sections: 'Log on' and 'Not a member yet? Sign up now!'. The 'Log on' section includes fields for 'User name:' and 'Password:', a 'Remember me' checkbox, a 'Log on' button, and a 'Forgotten password' link. The 'Sign up now!' section includes fields for 'First name:', 'Last name:', 'E-mail:', 'Password:', 'Password strength:', and 'Confirm password:', along with a 'Register' button. At the bottom of the page, there is a footer that reads 'This is a sample web site for Kentico CMS'.

3. Log in as the administrator again or try registering a new account. After you sign in successfully, the site automatically redirects you back to the **Partners** page.

The screenshot shows the secured page for partners. The page has an orange header with the company name and a blue navigation bar with links for Home, News, Partners, Products, Services, and Search. Below the navigation bar, there is a status bar that reads 'Current user: Global Administrator (administrator) Sign Out'. Below the status bar, there is a message that reads 'This is a secured page for partners.' At the bottom of the page, there is a footer that reads 'This is a sample web site for Kentico CMS'.

Here you can see the content of the secured page, as well as the name of the current user and the **Sign Out** button.



### Displaying content based on user permissions

Kentico CMS also allows you to display content according to the read permissions of users. For example, you can grant the Read permission for a Gold partners section to members of the *Gold partners* role, so that only gold partners are able to see the corresponding menu item and page content.

You can find more information about permissions in the [Development -> Membership, permissions and security](#) chapter of the Developer's Guide.

This concludes the creation of the sample website.

**Part**

---

**VIII**

**Further steps**

---

---

## 8 Further steps

### 8.1 Further steps

This is the end of the Kentico CMS Tutorial. If you need any further details, you will find them in [Kentico CMS Developer's Guide](#). It also covers other advanced topics, such as:

- Multi-lingual content
- Multi-site configuration
- Workflow and versioning
- Security administration
- Deployment to the live website
- Newsletters, Forms and other modules
- Kentico CMS API and extensibility
- and many other features.

If you cannot find the information that you require, please feel free to contact us at <http://www.kentico.com/Support.aspx>.

# Index

## - A -

- App themes 58
- ASPX page template
  - creating 39
  - editing in the portal engine 48
  - overview 37
  - registering the page as a template 42
  - web parts and widgets 48
  - writing the code 41

## - C -

- CSS styles 56

## - F -

- Further steps 123

## - H -

- home page
  - editing content 20

## - I -

- image
  - inserting 25
- installation
  - setup 8
  - web application 9

## - K -

- Kentico CMS Overview 5

## - L -

- link 26

## - M -

- master page
  - using 45
- menu design
  - overview 59
- minimum requirements 7

## - N -

- new page 21
- new site
  - configuring the web project 71
  - creating the CSS stylesheet 68
  - home page 78
  - main menu 77
  - master page 72
  - news page 87
  - products 97
  - products - new document type 97
  - products - page template 106
  - products - transformations 103
  - search page 110
  - secured section 115
  - services page 92
  - wizard 63
- new site - ASPX templates 63
- news item 28

## - O -

- Overview 5

## - P -

- prerequisites 7

## - S -

- Setup 11
- Site Development Overview 32
- Support 5, 123
- system requirements 7

---

## - T -

Technical support 5, 123

## - U -

User interface overview 17

## - W -

web installer 9

widgets and web parts on ASPX templates 48