Kentico 8.2 Tutorial

Tutorial - Developing websites with Kentico 8.2

Welcome to the Kentico Tutorial!

This section of the documentation explains the basics of editing content in Kentico, shows you how to create web pages, and walks you through the development of a simple website.

We recommend that you read the tutorial from the beginning to the end, to make sure you don't miss any step in the development process.

Before you begin

Before you dive into the tutorial, you must ensure that you have Kentico installed on your local machine, or on a remote server, including the sample **Corporate site**.

If you already have Kentico installed, but don't know how to install the sample Corporate site, see Installing new sites, then Creating new sites from templates.

Start learning Kentico

After you've installed Kentico, go to the first page of the tutorial and start learning.

Using the Kentico interface

To access the Kentico administration interface:

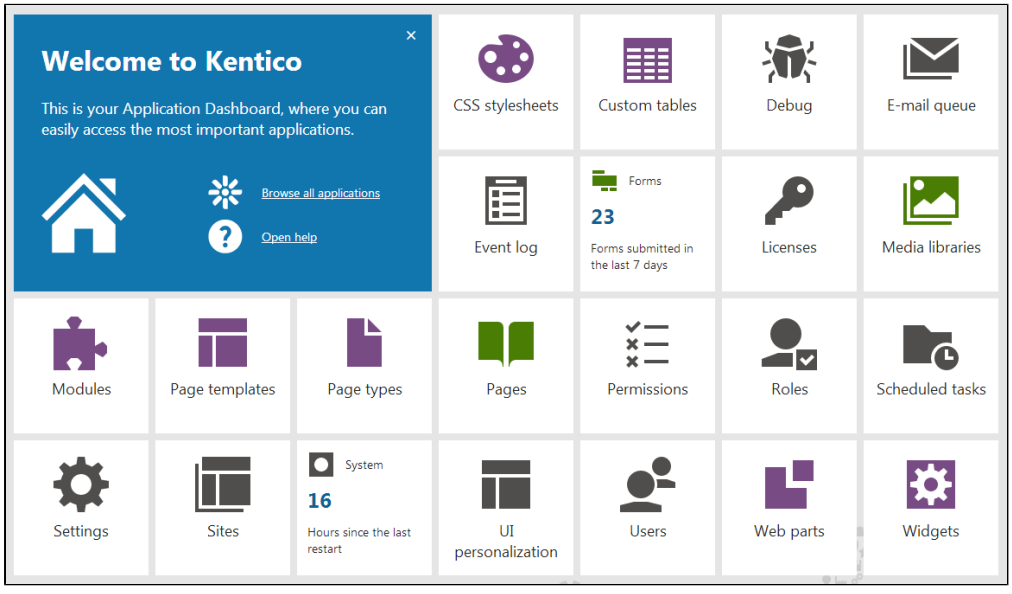
- Type **/Admin** after your site's domain into the browser address bar (for example *http://mysite.com/admin*)
  OR
- Click the **Administration** link at the top of the sample Corporate site.

A logon page appears where you need to enter a username 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 to a live server.

Once you log in, the Kentico administration interface opens. The interface is separated into ***applications***. Each application represents a group of related functions. Applications allow you to perform various types of tasks, configure the system, and view information.

The first page that you see after logging in is the ***application dashboard***. The dashboard provides quick access to your most frequently used applications. Certain application tiles in Kentico work as *Live tiles.* This means that they provide information that may require the user's attention by displaying application data.



Adding applications to the applications dashboard

Each user in Kentico can individually configure the applications they see on their application dashboard.

1. Click the **Edit dashboard** icon.
2. Click on **Add new application** (plus sign). The application list opens.
3. Select the applications that you want to pin to the application dashboard.
4. Confirm by clicking the **Edit dashboard** icon again.

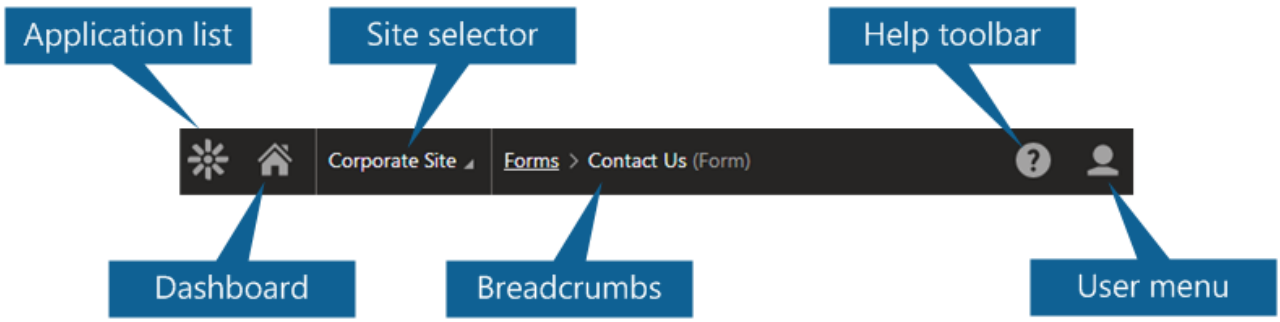
> **Predefining the application dashboard for roles**
> Administrators can also predefine the application dashboard for particular roles.

Moving applications on the application dashboard

To move applications on the dashboard, simply drag & drop them to a new place.

Administration interface

All parts of the administration interface share the same header, which contains the following elements:



Application list	Opens a list of all applications that you are allowed to access, divided by category. Use the search bar to quickly find the application you need, or browse the categories. You can move between applications by pressing the up and down arrows on your keyboard. Press Enter to open the selected application. Click **Live site** at the bottom of the list to open a new browser tab with the live version of the current website. Keyboard shortcut: F2
Dashboard	Returns you to the application dashboard. Keyboard shortcut: F4
Site selector	Allows you to switch between different sites if your Kentico installation contains multiple sites.
Breadcrumbs	Displays the current application, and your location within the application's interface. You can click on items in the breadcrumbs to navigate back within the interface structure.
Help toolbar	Provides links to relevant pages in the main Kentico documentation (based on the current application). Keyboard shortcut: F1
User menu	Expands a menu where you can adjust the settings and preferences of your user account. Also allows you to **Sign Out** of the administration interface, which takes you to the live website as an anonymous visitor.

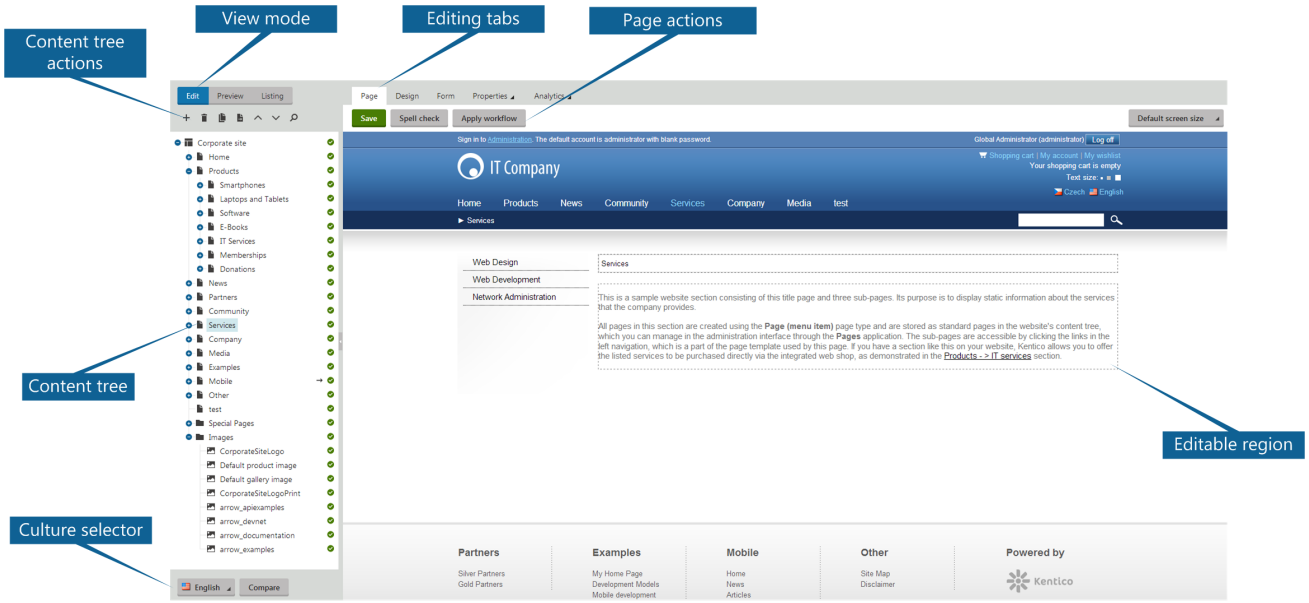
> **Tip**: You can open any number of applications at the same time in different browser tabs. Right-click the application in the dashboard or application list, and click Open link in new tab.

Managing content - The basics

In this section, you will learn the basics of editing web pages in Kentico.

> Please make sure you have the sample **Corporate Site** installed before you continue.

To manage the content of websites, log in to the Kentico administration interface and open the **Pages** application. Here you can edit the site selected in the interface's main header.

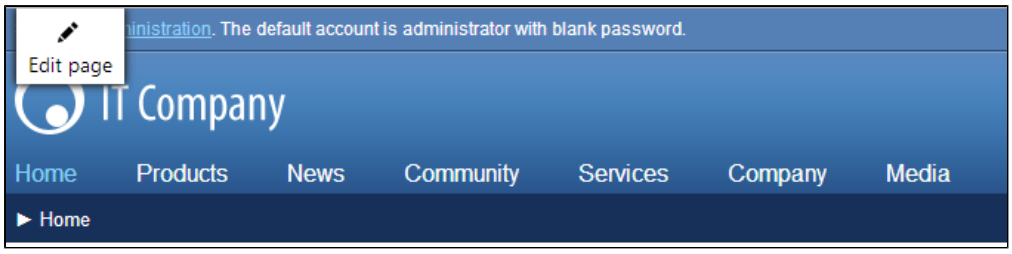


The **Pages** application consists of the following sections and features:

- **Content tree** that represents the site map of the website. Use the tree to organize the site's structure and select specific pages.
- Toolbar with **Content tree actions** that allow you to create, delete, copy and move pages.
- **View mode** selector where you can switch between edit, preview and listing modes.
- **Editing tabs** used to move between editing of page content, designing page templates (for developers), editing page fields, configuring page properties, and monitoring of page analytics.
- **Culture selector** that allows you to switch between different versions of multilingual websites.
- When editing pages, you can use the **Text editor** and **Page actions** to add and format page content (text, images, advanced dynamic content).
    - Available for **Editable regions** on the **Page** tab and when editing pages fields on the **Form** tab.

On-site editing

In addition to the main administration interface, Kentico also provides a way to edit page content directly while browsing the live website. Authorized editors can access on-site editing mode by typing **/Admin** after the site's domain into the browser address bar, or by clicking the **Edit page** button in the corner of pages.



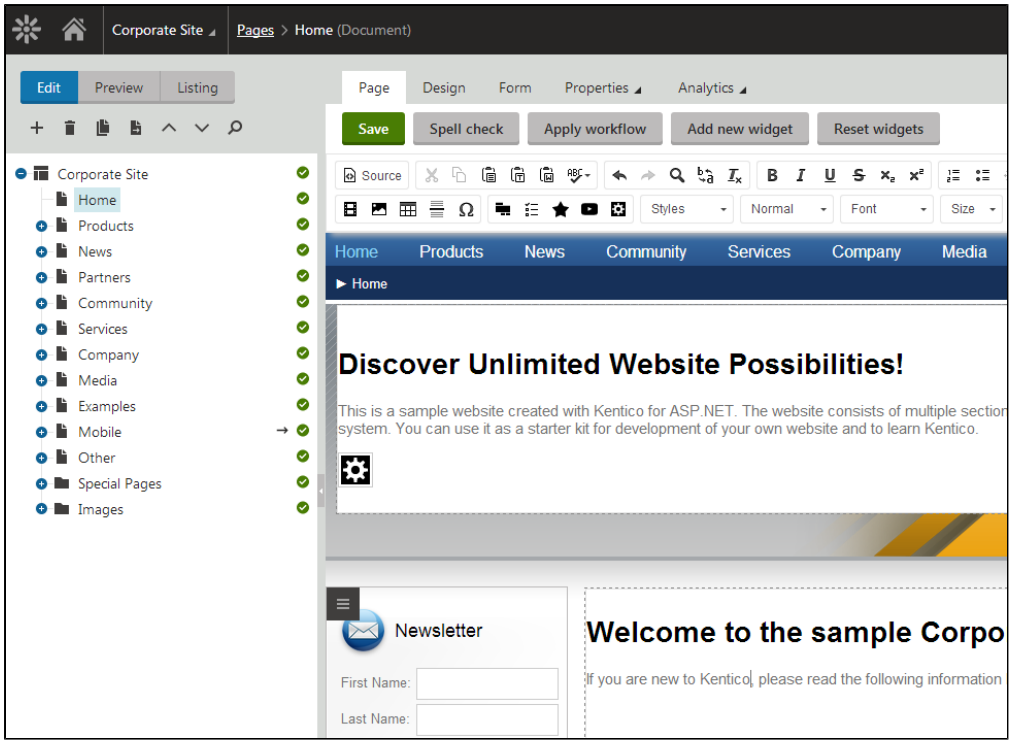
All actions available in on-site editing mode can also be done through the **Pages** application. The on-site editing mode simply provides an alternative way to edit websites.

This tutorial demonstrates all operations in the Pages application — we recommend first becoming familiar with the **Pages** application and the general structure of website content before you use on-site editing.

Editing home page content

To modify the content of the Corporate site's home page, open the **Pages** application in the Kentico administration interface.

Click the **Home** page in the content tree.



You can see two editable regions on the **Page** tab (in the **Edit** view mode). Editable regions have a dotted outline. Delete all content from one of the regions and enter the following text instead:

> This is my first text.

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

> This is my first text.

Click **Save** above the editor toolbar or press **CTRL+S** to save the changes.

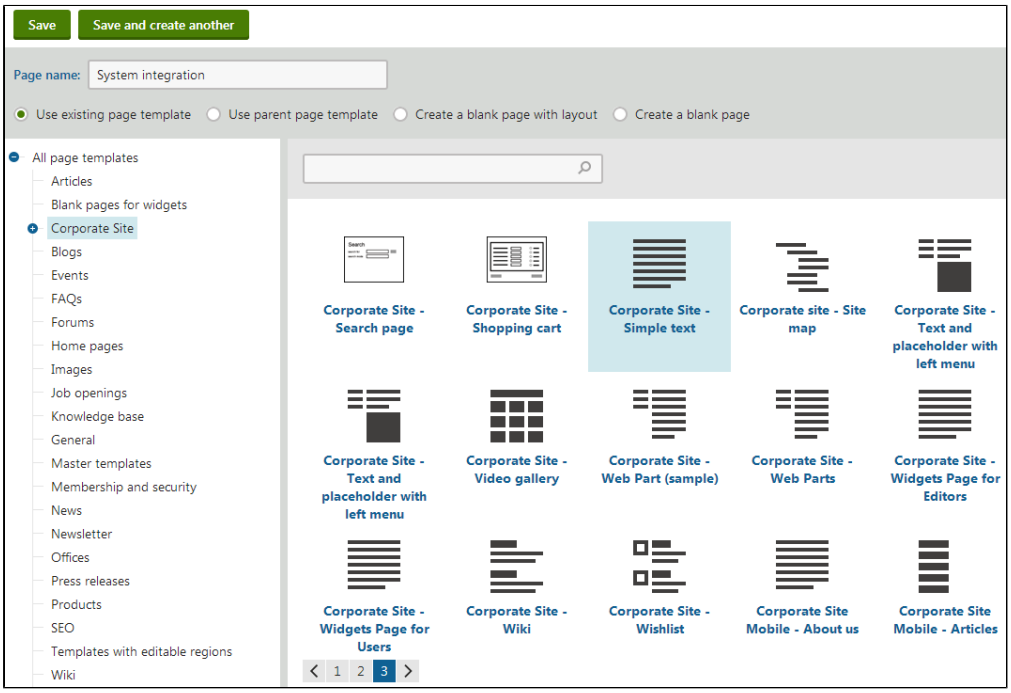
Now switch to **Preview** mode to see the modified version of the home page as it appears to site visitors.

Creating a simple page

Now create a new page under the website's **Services** section.

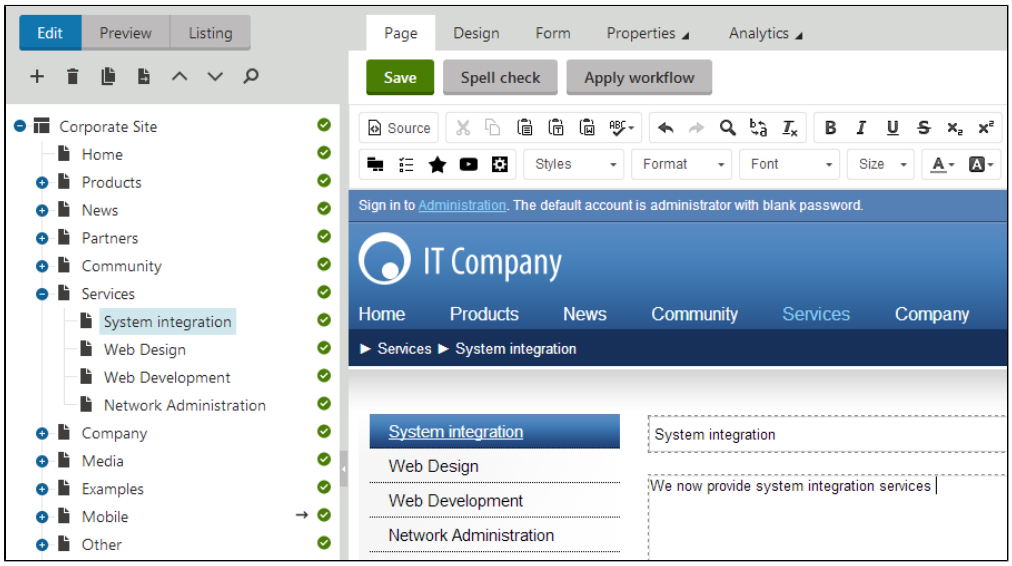
1. In the **Pages** application, return to **Edit** mode.
2. Click **Services** in the content tree.
3. Click **New** (+) above the content tree.
4. Select the type of the page you want to create under the selected page. Click the **Page (menu item)** option.

5. Type ***System integration*** in the **Page name** field.
6. Choose the **Corporate Site -> Corporate Site - Simple text** template:



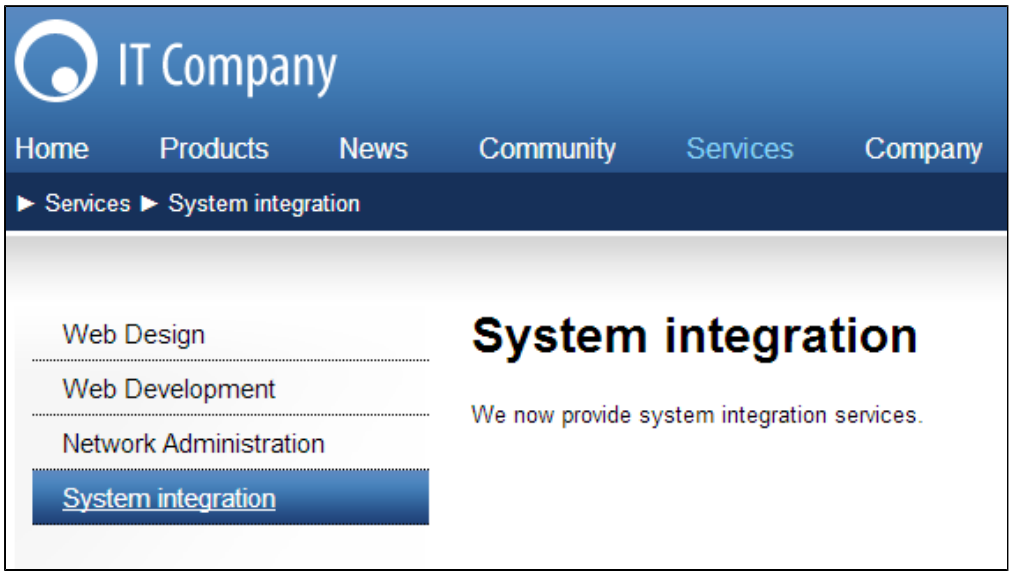
7. Click **Save** to create the new page.

The page appears in the content tree. You can add content on the **Page** tab. Type some text in the editable regions and click **Save** again.



You can change the order of pages in the content tree (and on the website). To move the **System integration** page to the end of the Services section, select the page and click **Move down** ( ) above the content tree.

Switch to **Preview** mode. You can now see the new page as it appears to site visitors.

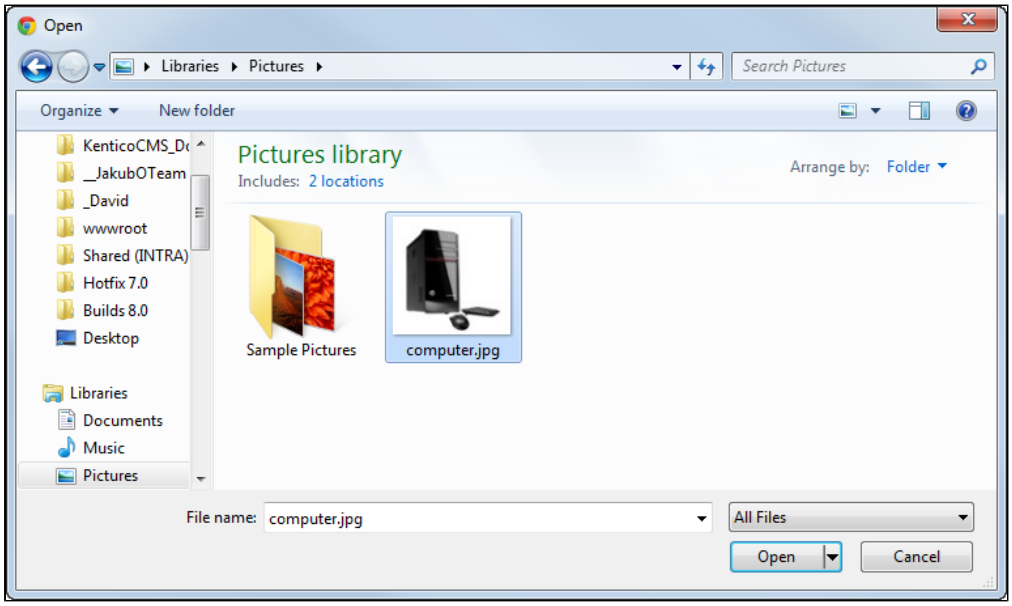


Inserting an image

Now upload and insert an image onto the **System integration** page created in the previous section.

1. In the **Pages** application, return to **Edit** mode.
2. Select **Services -> System integration** in the content tree.
3. On the **Page** tab, place the cursor into the main editable region, just below the text, and click **Quickly insert media** on the text editor toolbar.

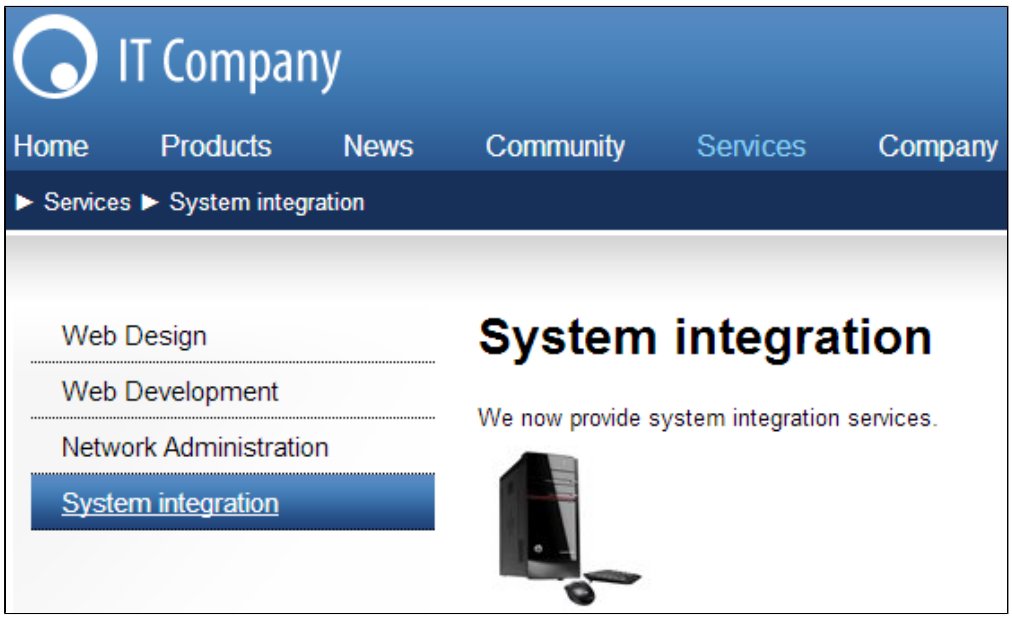
4. Your browser's **Choose file** dialog opens. Locate a suitable image file and click **Open.**



> **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 to see the file selection dialog. This applies only to the administration interface (site visitors are not affected).

5. Click **Save**.

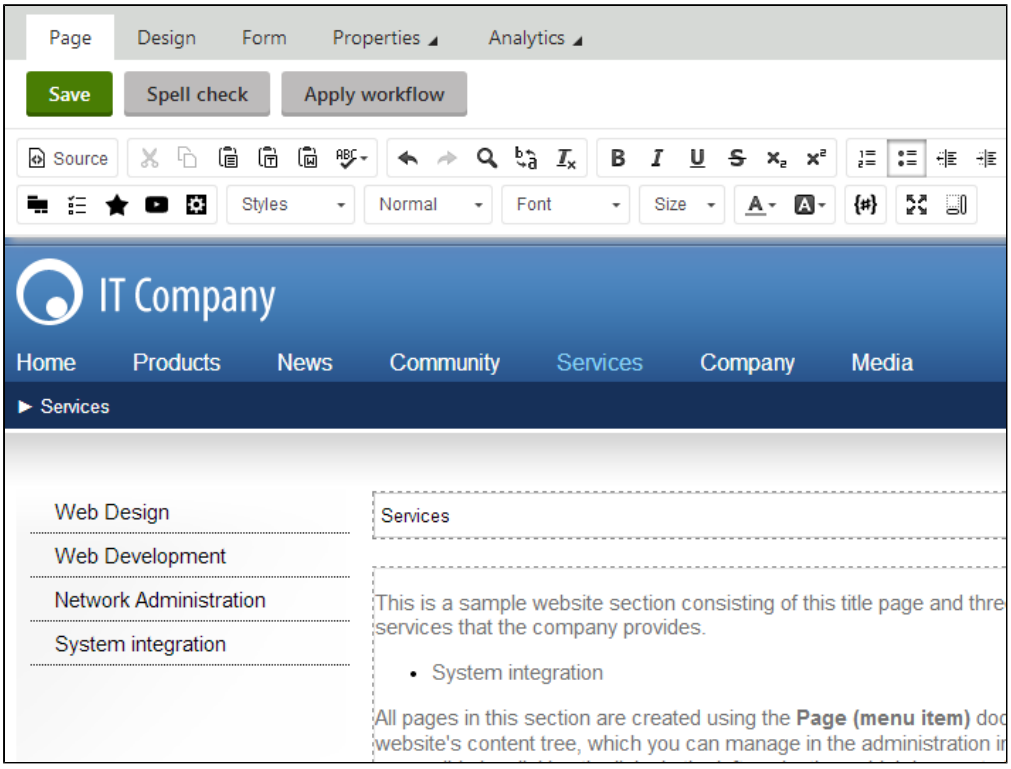
The image is now saved in the editable region and appears on the page. Switch to **Preview** mode to view the modified appearance of the page.



Inserting a link

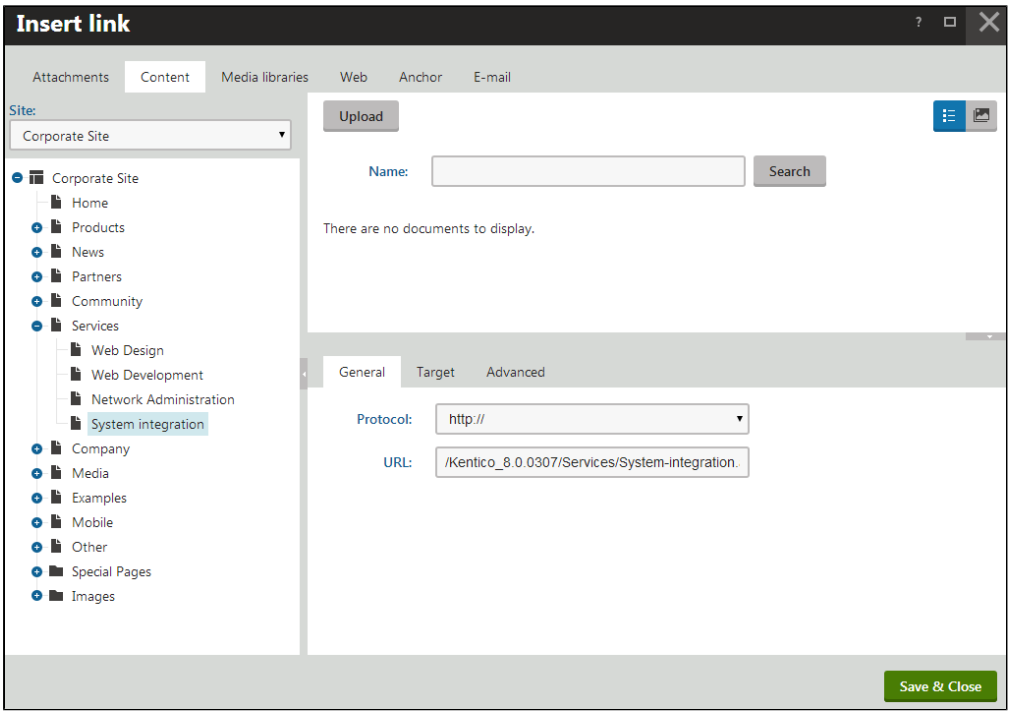
Now create a link between the **Services** page and the new **System integration** page.

1. In the **Pages** application, return to **Edit** mode.
2. Select **Services** in the content tree.
3. On the **Page** tab, add a bulleted list item into the second editable region and type **System integration**.



4. Select the whole line and click **Insert/Edit link** on the text editor toolbar.

5. The **Insert link** dialog opens. On the **Content** tab, select the **Services -> System Integration** page and click **Save & Close**.



6. Click **Save** on the **Page** tab.

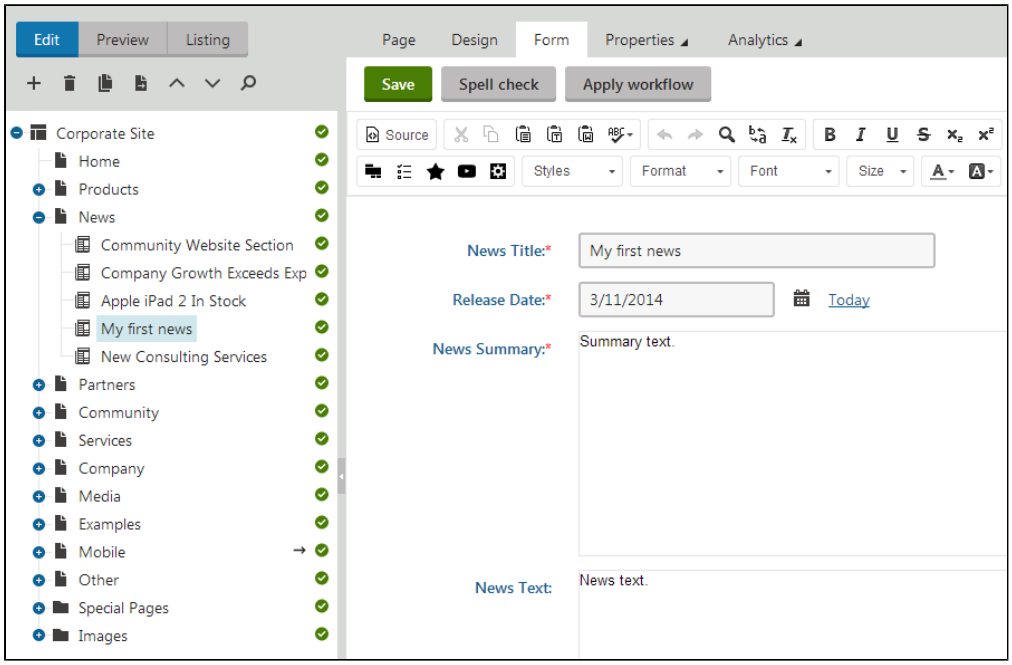
The text is now marked as a link (underlined). Switch to **Preview** mode and click the **System integration** link. The new page opens in the preview section.

Creating a news article

Now create a news article on the website.

1. In the **Pages** application, return to **Edit** mode.
2. Select **News** in the content tree.
3. Click **New** (+). The system automatically chooses the **News** page type (the only type allowed in the News section).
4. Fill in the news page fields in the editing form:
    - **News Title**: My first news
    - **Release Date**: click **Today**
    - **News Summary**: Summary text.
    - **News Text**: News text.
5. Click **Save** to create the new page.

The news page appears in the content tree (and on the website).



When editing news pages, you use the **Form** tab instead of the **Page** tab. This means you are not editing the content of editable regions, but rather the structured data fields of the given page. The page fields are fully customizable for every page type.

If you switch to **Preview** mode, you can see the data of the news page displayed on both the **News** page (list) and **News -> My first news** page (details).

Page versus form

Pages have two possible types of content:

- Content stored in editable regions on the page
- Data stored in form fields

The following table compares both approaches:

	Editable regions on the Page	Form
Content structure	Simple content structure.	Complex content structure, typed data, such as text, date-time values, numbers, files etc.
Validation	Only supports basic validation rules for minimum and maximum length.	Customizable validation rules, including regular expressions and custom form controls with custom validation code.
Display	The content is displayed on the page, just like it appears in editing mode.	You need to use listing web parts or controls to display the content using transformations.
Storage	The content of all editable regions is stored in a single XML field in the data of each page.	The content is stored in a separate database table for the given page type. Each field has its own column. The data can be easily modified using SQL queries or the API.
Examples of use	Home page, contact page. Generally: pages with simply structured or unstructured text-based content. The editable regions are usually used for pages of the **Page (menu item)** type.	News, product specification, event details, job openings, etc. Generally: pages with structured content where you need to separate content from design and keep the content in its original data type. Form-based content is usually used for pages types such as **News, Product, Article**, **Blog**.

Site development overview

Kentico provides two basic development models. You can choose the approach that best suits your needs:

Portal engine	Recommended for most developers. Does not require programming and using Visual Studio. You can build websites using page templates and web parts in a browser-based user interface. To learn the basics of the portal engine, see Creating pages using the portal engine, then Walkthrough - Creating a new site using the Portal engine
ASPX templates	Can be chosen by ASP.NET developers who prefer to create websites using standard ASP.NET architecture and 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. To learn the basics of ASPX template development, see Creating pages using ASPX templates, then Walkthrough - Creating a new site using ASPX templates.

Both approaches are fully supported and 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.

If you're not sure which development model is best for you, see Choosing the right development model.

Creating pages using the portal engine

The Kentico Portal engine allows you to create dynamic web pages without any programming knowledge. With the Portal engine, you don't need to use Visual Studio or any other web development tool. Instead, you create reusable page templates directly in your web browser.

What you need to use the Kentico Portal engine:

- basic knowledge of HTML and CSS to be able to create page layouts
- a supported web browser

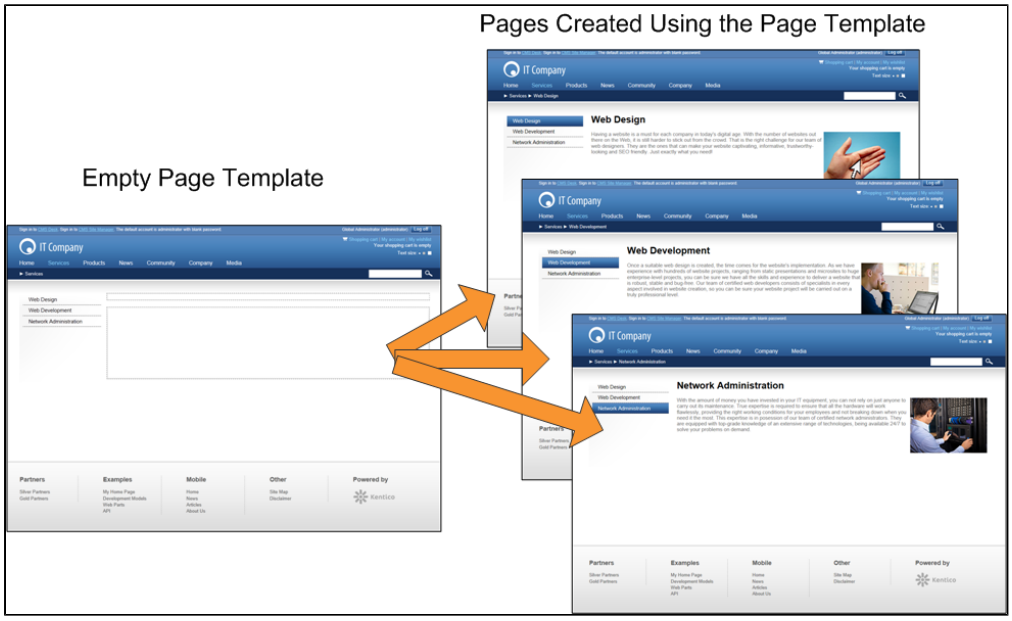
Start by learning what Page templates are and how you can use them.

Page templates

The Kentico portal engine provides a highly flexible and extensible framework that allows you to build complex dynamic websites using built-in or custom web parts.

What is a page template?

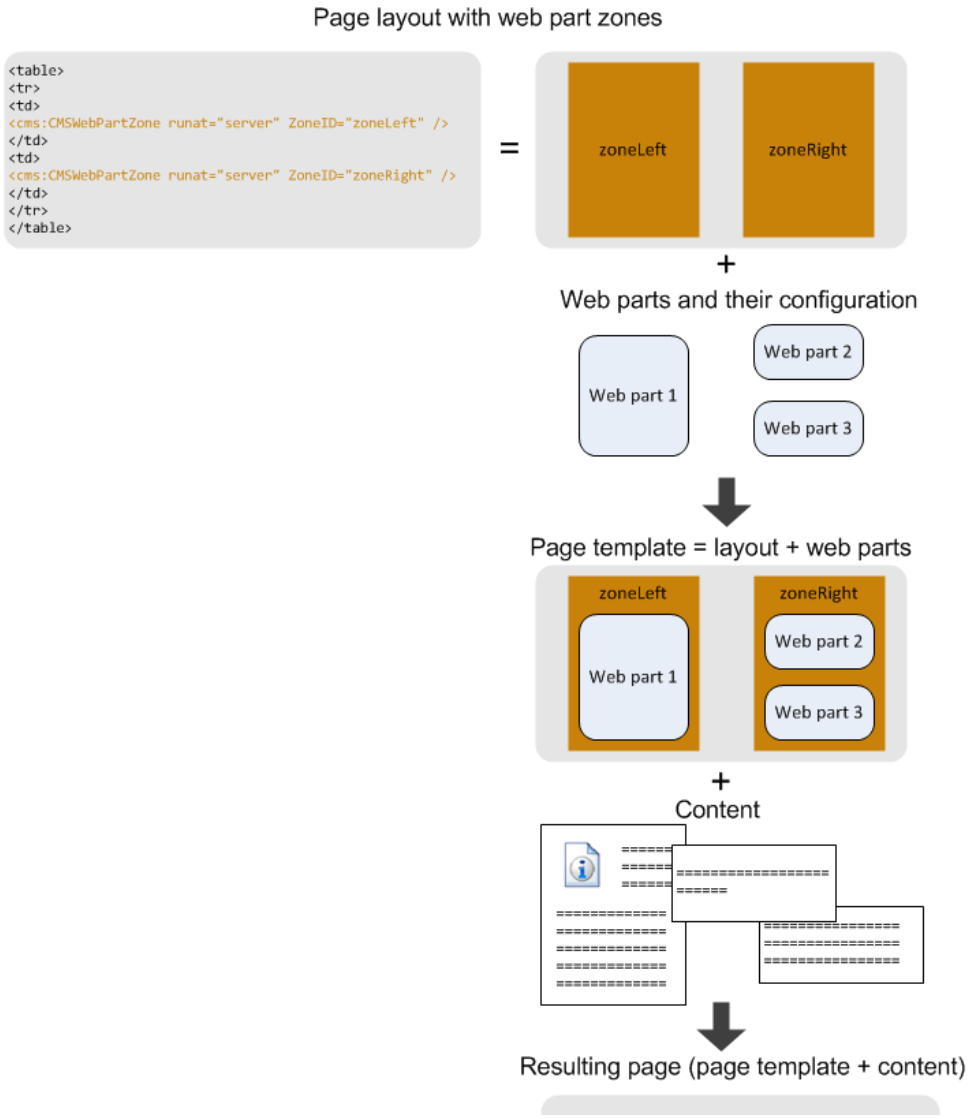
Every web page is based on a page template. The page template can be specific for a single page (so called "ad hoc" page template) or re-used for any number of pages. The following picture shows examples of pages that use the same page template.

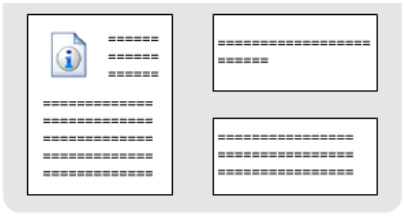


The pages have different content, but use the same header, menu content structure, and footer — they are based on the same page template. Templates allow you to quickly create multiple pages with the same design.

What does a page template consist of?

Portal engine page templates are a combination of a **page layout** (ASCX or HTML code) and **web parts**. The following figure illustrates the structure of page templates and how they are used to display pages.


Page layout with web part zones
<table>
<tr>
<td>
<cms:CMSWebPartZone runat="server" ZoneID="zoneLeft" />
</td>
<td>
<cms:CMSWebPartZone runat="server" ZoneID="zoneRight" />
</td>
</tr>
</table>
=
zoneLeft
zoneRight
+
Web parts and their configuration
Web part 1
Web part 2
Web part 3
Page template = layout + web parts
zoneLeft
Web part 1
zoneRight
Web part 2
Web part 3
+
Content
Resulting page (page template + content)



The **page layout** is a piece of fully customizable HTML code that defines the design of the page and contains **web part zones**. The web part zones represent areas where designers can place **web parts**. The web parts display page content or provide functionality, such as input forms. The layout and the web parts together define the **page template**.

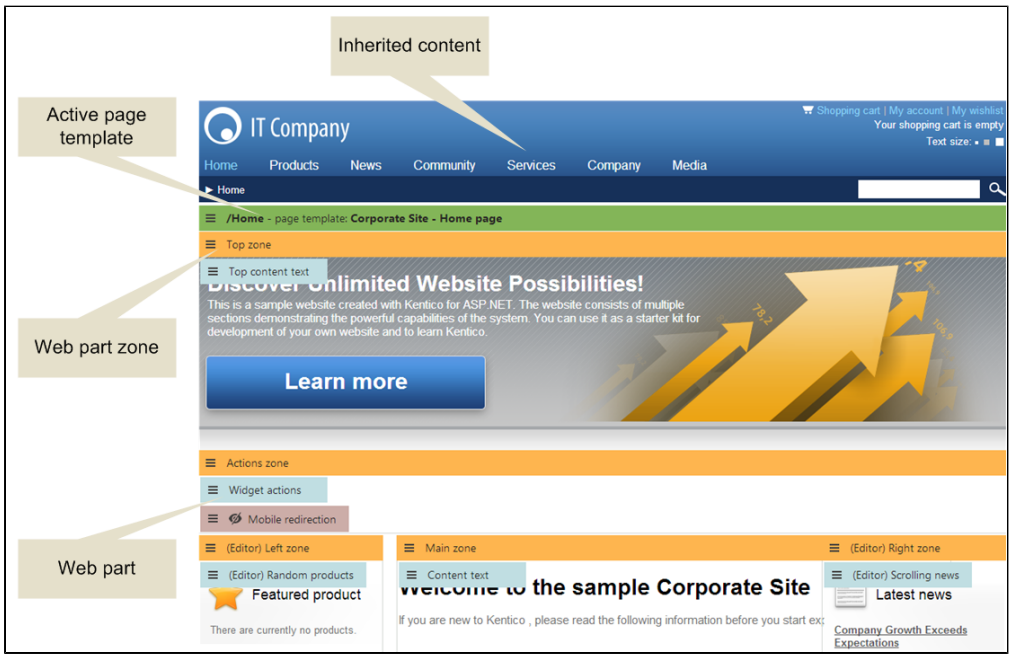
When you add **page specific content** to the page template, you get the **final page**.

Modifying page templates

Now you will learn how to work with the layouts of portal engine page templates.

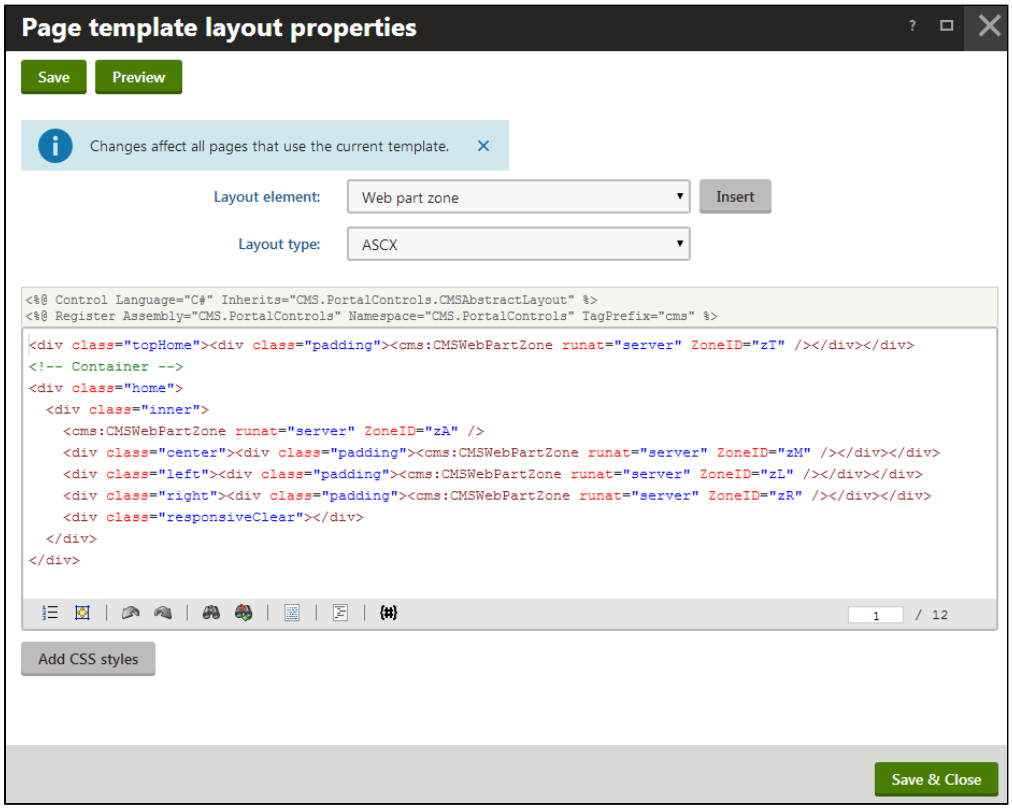
1. Open the **Pages** application.
2. Select the **Home** page in the content tree.
3. Switch to the **Design** tab.

Here you can see the the structure of the home page template. The page header and main menu are inherited from the root page. The "active" area, designated by a green header, contains the orange web part zones and instances of web parts.



The page is based on the page template ***Corporate Site - Home page***. The page template consists of web parts that are placed inside web part zones. The placement of the web part zones is specified by the template's page layout.

To modify the layout of the template, click the menu icon ( ) in the green header of the active page template and select **Edit layout** in the menu.



The page layout consists of basic HTML (or ASPX) code. It contains **CMSWebPartZone** controls, which define the web part zones on the template. Each zone control must have a unique ID. You can format the layout using any type of HTML code — it's up to you whether you use tables or a CSSbased layout.

Working with the layout code

You can use the selector above the code editor to quickly **Insert** web part zones and other advanced layout elements.

By clicking **Preview** in the header of the editing dialog, you can edit the layout code side-by-side with a preview of how the changes affect the live site version of the page.

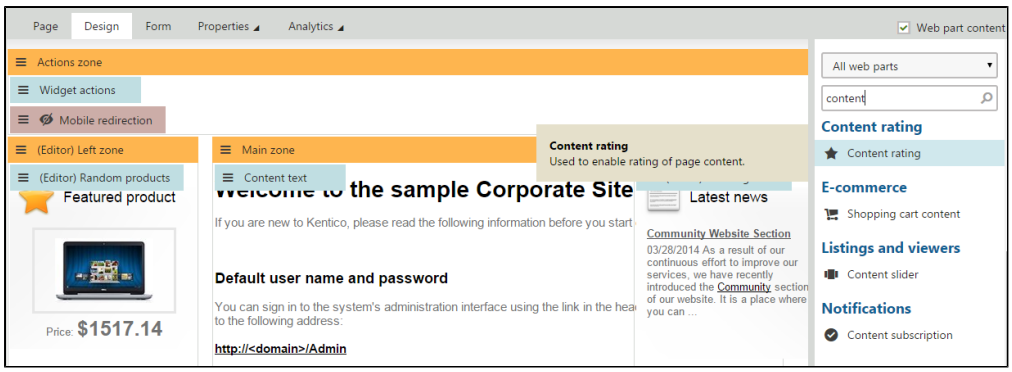
> **Access control**
>
> Only users who have the *Global administrator* **Privilege level** or have the **Design web site** permission assigned for the Design m odule can open pages on the **Design** tab and edit page layouts. This functionality cannot be accessed by regular content editors.

Modifying the home page structure

Now you will learn how to define the content of a page template's web part zones. Specifically, we will add a web part that allows visitors to rate content onto the template used by the Home page.

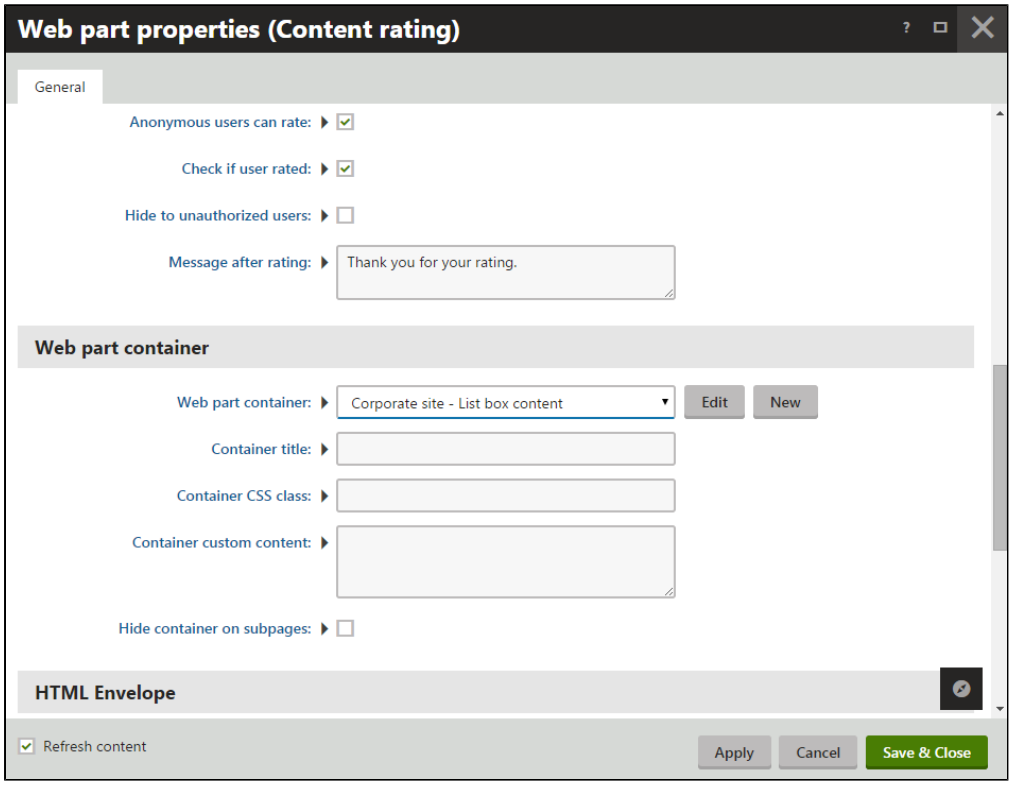
1. Make sure you are in the **Edit** mode of the **Pages** application.
2. Select **Home** in the content tree.
3. Open the **Design** tab.

You can insert web parts using the toolbar displayed on the right side of the tab. For example, type the word "content" into the search text box at the top of the toolbar. This limits the listed web parts to those that have the word "content" in their name.



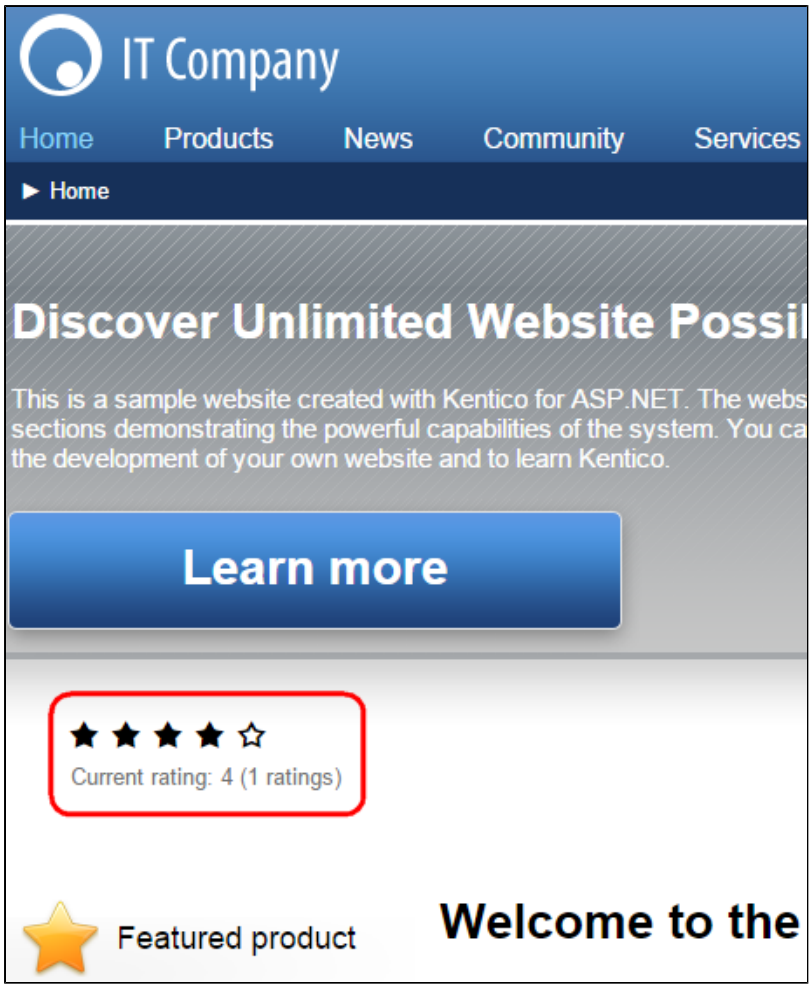
Hover over the **Content rating** web part, hold down the mouse button and drag it from the toolbar to the bottom of the **Actions zone**, below the **Mobile redirection** web part. After you drop the web part into the zone, the **Web part properties** dialog opens.

Scroll down to the **Web part container** property and select ***Corporate Site - List box content*** as the value.



Click **Save & Close** to save the changes. If you ever wish to adjust the properties of a web part at a later point, double-click the web part on the **Design** tab to open the **Web part properties** dialog again.

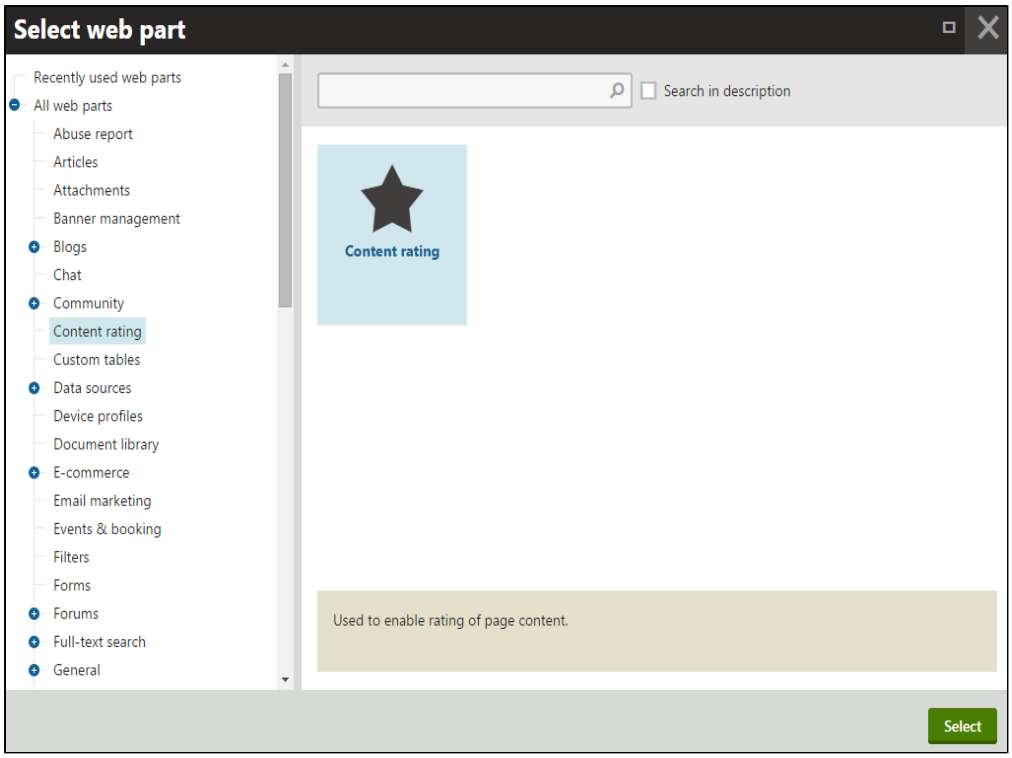
Switch to **Preview** mode to see how the new web part affects the appearance of the Home page.



Return to **Edit** mode and view the **Home** page on the **Design** tab again. The system provides an alternative way of adding web parts if you do not wish to use the toolbar.

Right-click the header of the zone where you want to insert the web part and select **Add new web part** in the menu.

The **Select web part** dialog opens, where you can look through the catalog of all available web parts. For example, the previously used **Cont ent rating** web part is in the **Content rating** category.



Close the dialog, since there is no reason to add the web part again.

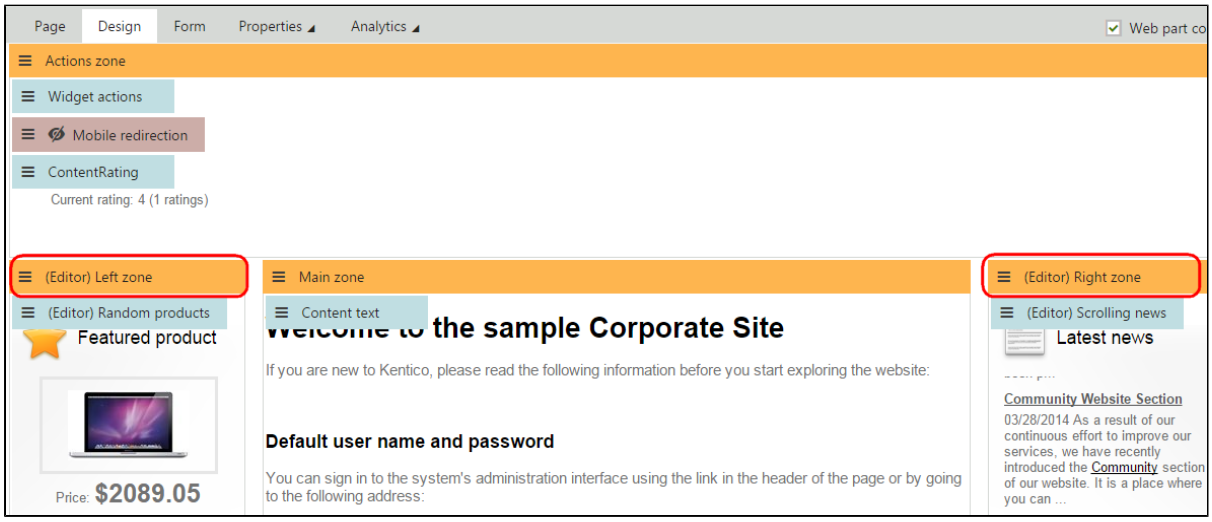
Both ways of adding web parts achieve the same result. When instructed to add a web part to a page template in the remaining parts of this tutorial, you can choose whichever method you prefer — either the toolbar or the zone action buttons.

> **Developing custom web parts**
>
> You are not limited to using the default set of Kentico web parts. You can find web parts created by other developers at the Kentico Marketplace or develop your own web parts.

> **Testing changes without switching the view mode**
>
> You can quickly test the changes you make to the structure of the website by opening or refreshing the given page in a different browser. Note that the view mode changes if you use two tabs of the same browser.
>
> Multiple tabs can however be useful for switching between different application in the administration interface.

Customizing pages using widgets

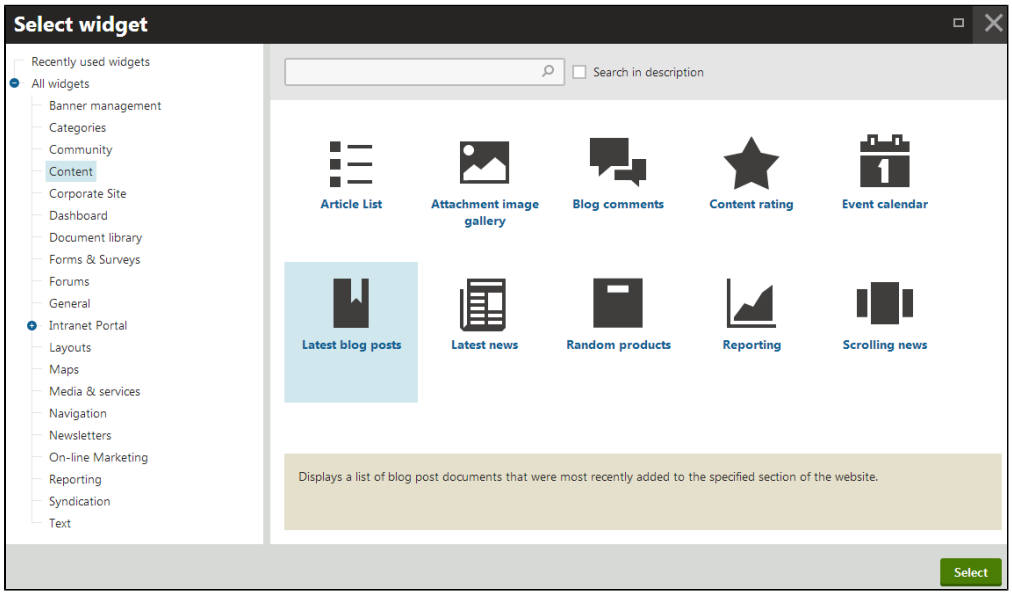
In the **Pages** application, view the **Home** page on the **Design** tab. The **Left zone** and **Right zone** zones have the **(Editor)** prefix before their name.



This identifies zones that contain widgets rather than web parts. Widgets are components placed into page template zones just like web parts, but they provide the option of page customization for various kinds of website users, not just administrators and designers. The two widget zones on the Home page are configured to allow customization by page editors. You can manage the content of the zones when editing pages on the **Page** tab of the **Pages** application.

Switch over to the **Page** tab, click the menu icon ( ) in the corner of one of the widget zones and then click **Add new widget** in the menu.

The **Select widget** dialog opens. The dialog is similar to the web part selection dialog, but with less items available. This is because every widget is based on an existing web part. Select the **Content -> Latest blog posts** widget and click **Select.**



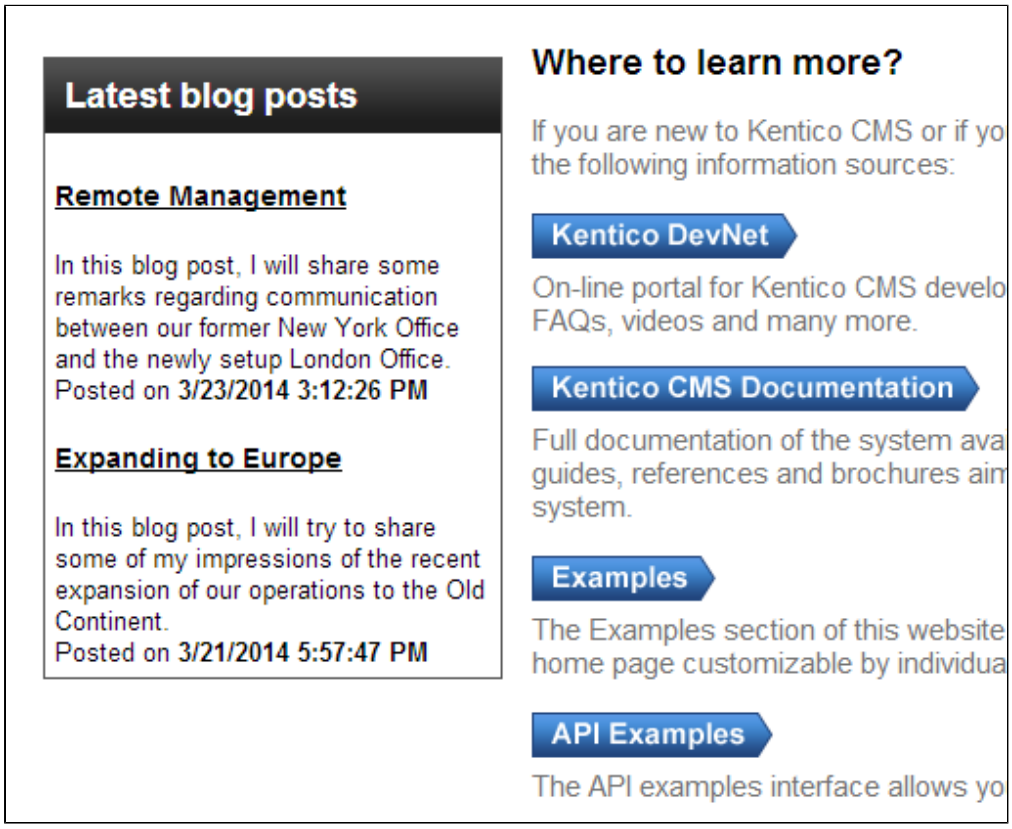
Enter the following values into the widget's properties:

- **Widget container**: Black box
- **Widget container title**: Latest blog posts

Leave the remaining properties in their default state and click **Save & Close**.

> The widget appears on the page, but is NOT permanently saved yet. When working with widgets on the **Page** tab, you need to click **Save** again on the page itself to confirm changes.

After you save the page, switch to **Preview** mode using the main toolbar to view the modified design of the home page. As you can see, content editors can alter pages with widget zones.

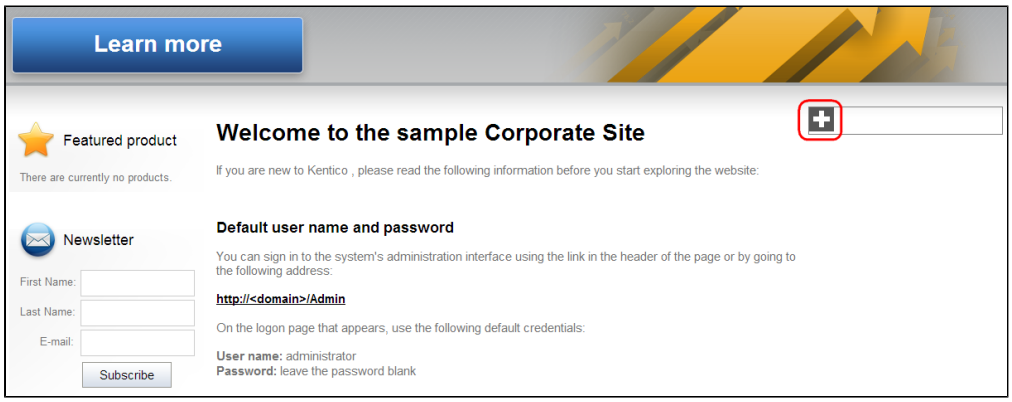


Now we will try out a different type of widget zone. Switch back to **Edit** mode and the **Design** tab. Expand the menu ( ) of the **Right zone** and click **Configure** in the menu.

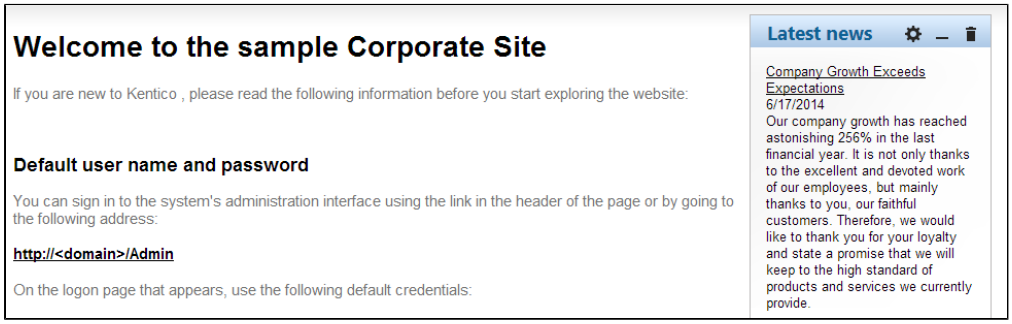
Set the **Widget zone type** property to **User personalization** and click **Save & Close**. The zone is now editable by ***registered users*** directly on the live site.

> **Note**: The system removes the content of a zone when you change the zone type.

View the live site (click the Kentico logo on the header to open the application list and click **Live site**). On the Home page, hover over the location of the right zone and click **Add new widget**.



Select the **Content -> Latest news** widget and click **Select**. In the properties dialog, leave the default values and click **Save & Close**. Changes made to widgets in user zones apply immediately — users on the live site do not need to save the page manually.



The content of user widget zones does not affect the global appearance of the page. Each user can only see their own personalized version of the user zones on pages. Log off the website and log in as a different user (enter the username **Andy** with a blank password). View the home page on the live site, and you can see that the zone is displaying the default content (empty in this case). The widget previously added under the administrator account is not visible. Every user can choose their own content for the zone from the available selection of widgets.

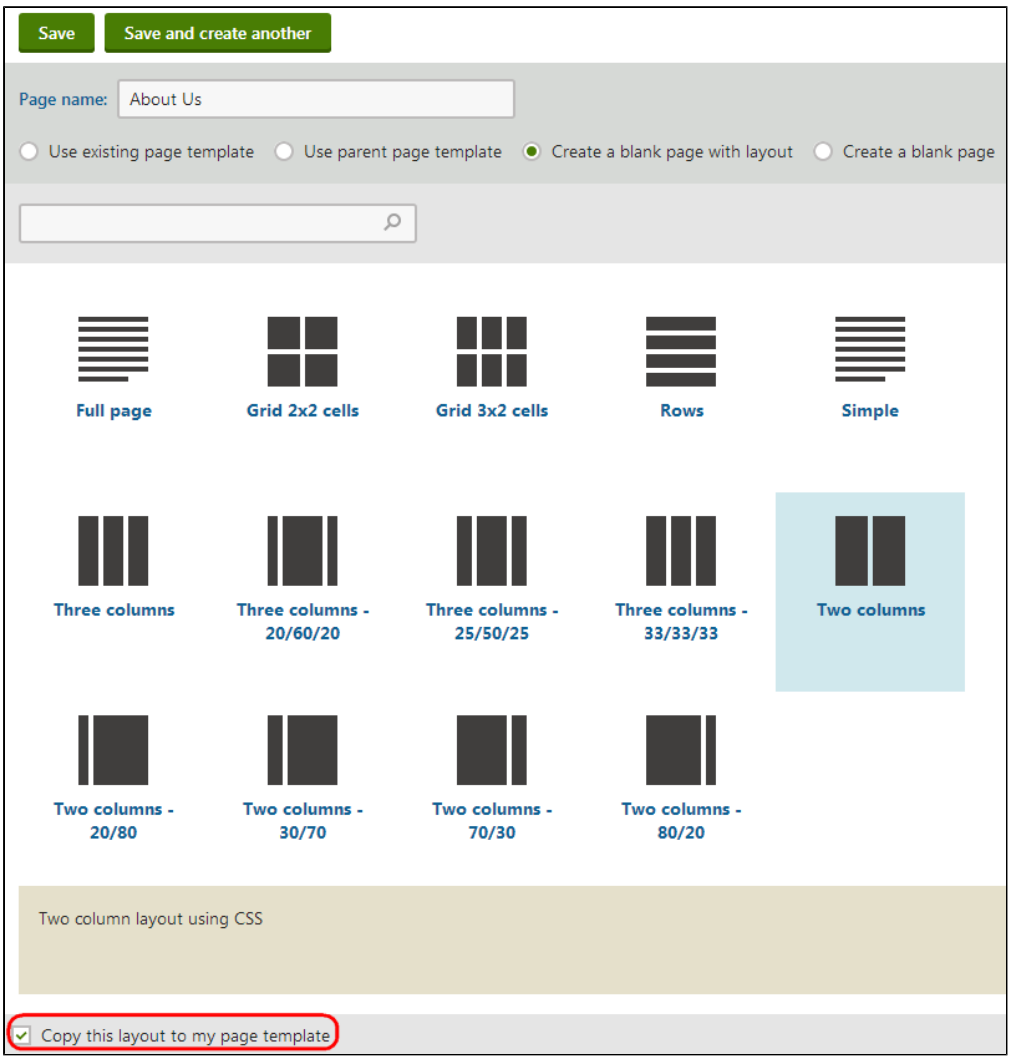
Log off and sign back in as the global administrator (username **administrator)**.

> **Managing widgets on the Design tab**
>
> You can add and configure widgets of all types directly on the **Design** tab of the **Pages** application. This sets the **default** content of widget zones for the template. The content displayed on pages does NOT match the default content once users make changes to the widget zones.

Creating new page templates

Now you will learn how to create a new page from scratch without using a predefined page template. We will create a new **About Us** page with two columns that contain editable regions.

1. Open the **Pages** application.
2. Select the root in the content tree and click **New** (+).
3. Choose the **Page (menu item)** page type.
4. Type **About us** as the **Page name**, choose to **Create a blank page with layout** and select the **Two columns** layout. Make sure that the **Copy this layout to my page template** box is checked — ensures that your page template uses a separate copy of the predefined layout without changing other pages.



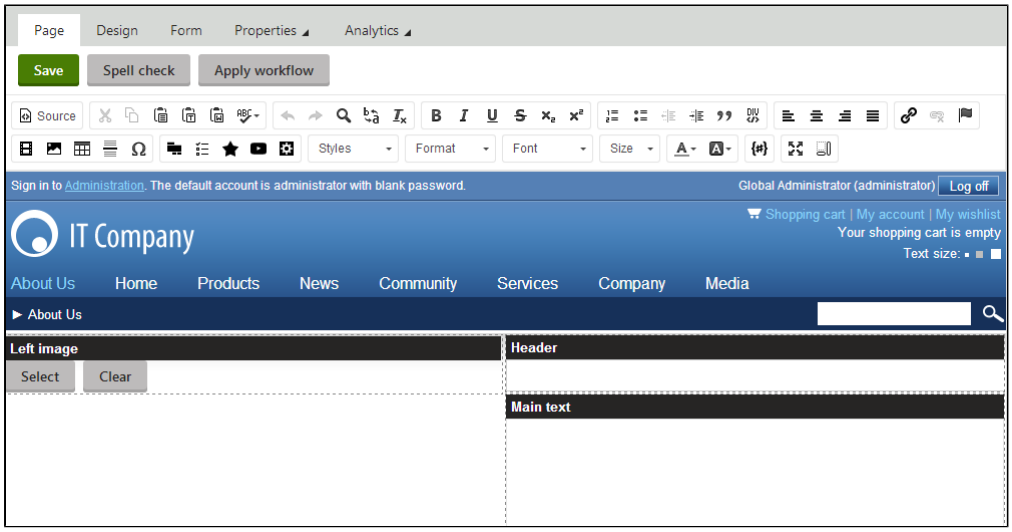
5. Click **Save** to create the new page. You cannot edit the page yet, because there are not editable regions on the new template.
6. Switch to the **Design** tab and add the **Editable image** web part to **zoneB**.
7. Configure (double-click) the web part and set the following properties:
    - **Web part control ID**: LeftImage
    - **Image title**: Left image

8. Click **Save & Close**.
9. Add the **Editable text** web part to **zoneC**. Configure the web part and set the following values:
    - **Web part control ID**: HeaderText
    - **Editable region title**: Header
    - **Editable region type**: Text box
    - **Content before**: <h1>
    - **Content after**: </h1>

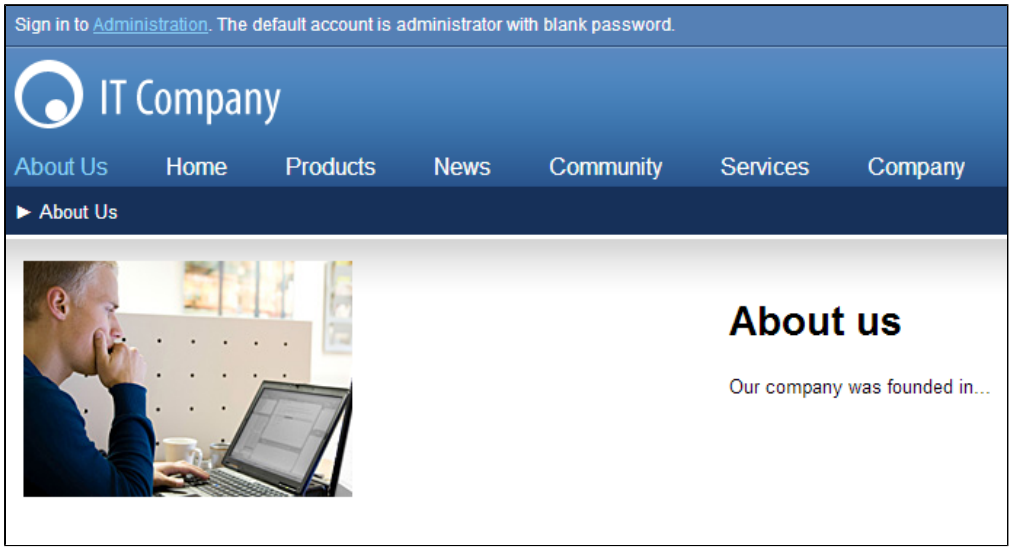
10. Add another **Editable text** web part to **zoneC** with the following properties:
    - **Web part control ID**: MainText
    - **Editable region title**: Main text
    - **Editable region type**: HTML Editor
    - **Editable region height**: 400

Result

Switch to the **Page** tab. The page now contains editable regions.



You can type in text content and choose an image using the **Select** button. Click **Save** and view the page in **Preview** mode.

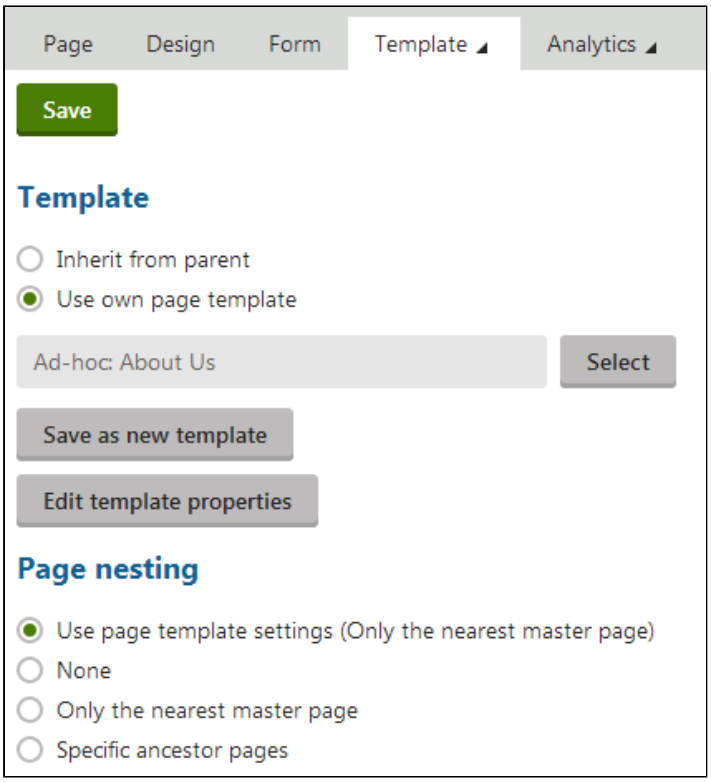


Re-using page templates

The previous tutorial page described how to create a new page with its own page-specific template. This type of page template is called an **a d-hoc page template**.

Now you will learn how to convert the ad-hoc template into a reusable page template, which can be used to create any number of pages with the same general layout, but different content.

1. Open the **Pages** application in **Edit** mode.
2. Select the **About Us** page in the content tree.
3. Switch to the **Properties -> Template** tab.



4. Click **Save as new template** and enter the following values:
    - **Template display name**: Left image with text on the right
    - **Template category**: Templates with editable regions
    - **Template description**: Two columns with an image on the left and text on the right.
    - **Assign to the current page**: Yes (leave the box checked)
5. **Save & Close**.

Result

If you try to add a new page now, the *Left image with text on the right* page template is available among the offered options in the **Templates with editable regions** category. You can create another page with the same structure as the **About Us** page.

Because the **Assign to the current page** option was checked, the system also automatically assigns the template to the current page instead of the previous ad-hoc template.

> **Important**: When you modify a re-usable page template, the changes affects all pages that use the template.

> When you delete a page which uses an ad-hoc page template, the template is automatically deleted together with the page.

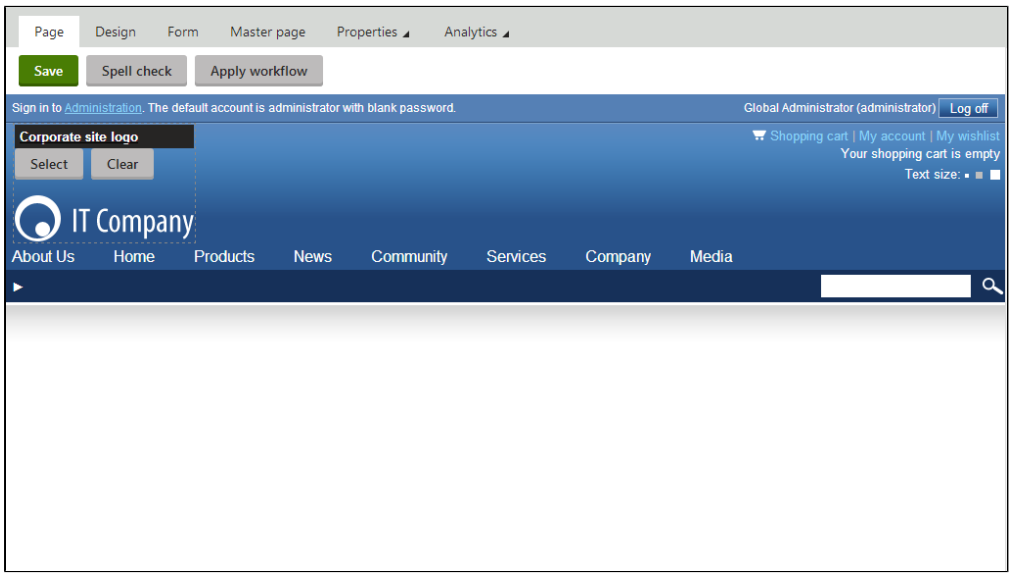
Content inheritance

Inheritance (page nesting) allows you to maintain a consistent design throughout the website and manage content shared by multiple pages in a single location. The portal engine implements inheritance by nesting subpages inside the content of ancestor pages on higher levels of the content tree.

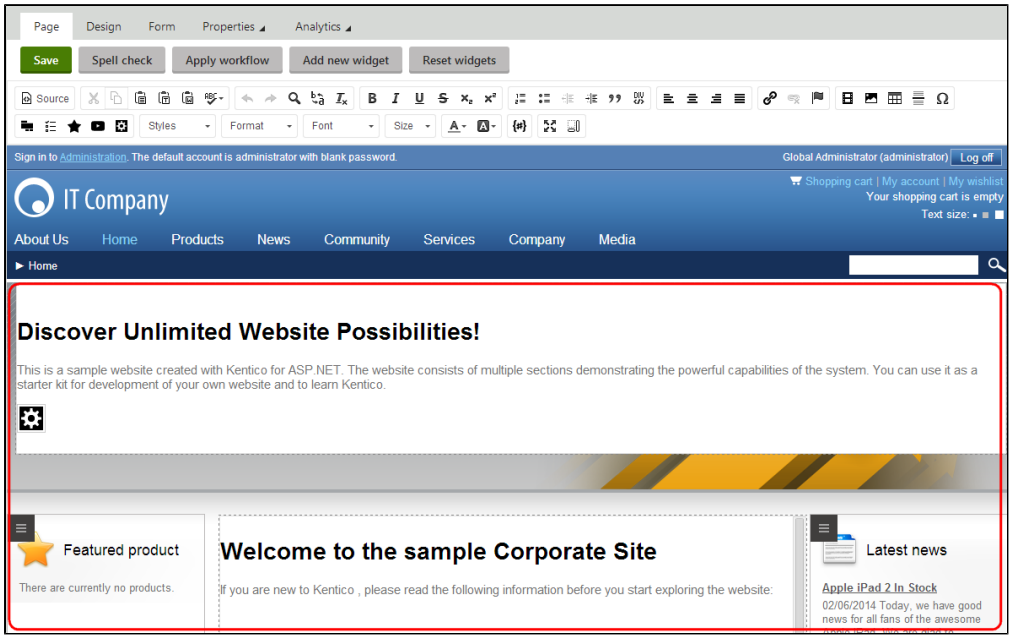
> **What are ancestor pages?**
>
> Ancestors include all pages under which a given page is stored, from the root of the site's content tree down to the page's direct parent.
>
> For example, the ***/Company/Offices/London Office*** page has the following ancestors:
>
> - Website root page
> - /Company

- /Company/Offices

Website root (master page) without nested content

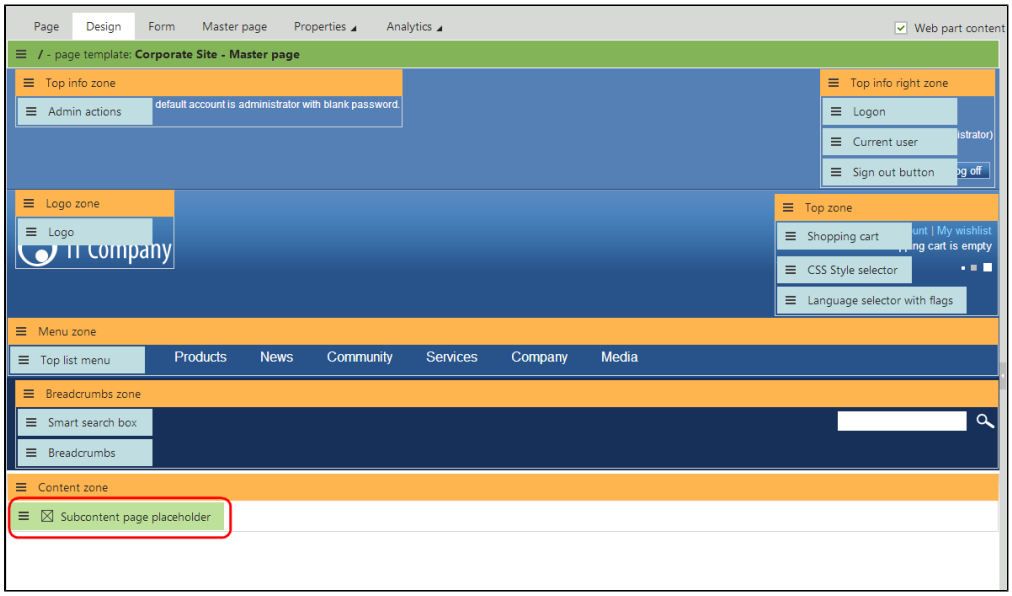


Home page nested inside the master page



Page placeholder

Note how the pages are structured in the content tree: the root (master page) is the parent of the Home page, which is loaded inside the master page. You can perform nesting recursively at any level of the content tree. A necessary component for page nesting is the **Page placeholder** web part. This web part must be placed on the master page, and specifies where the system inserts the underlying pages. The Page placeholder on the Corporate Site root loads the sub-pages between the main menu and footer.



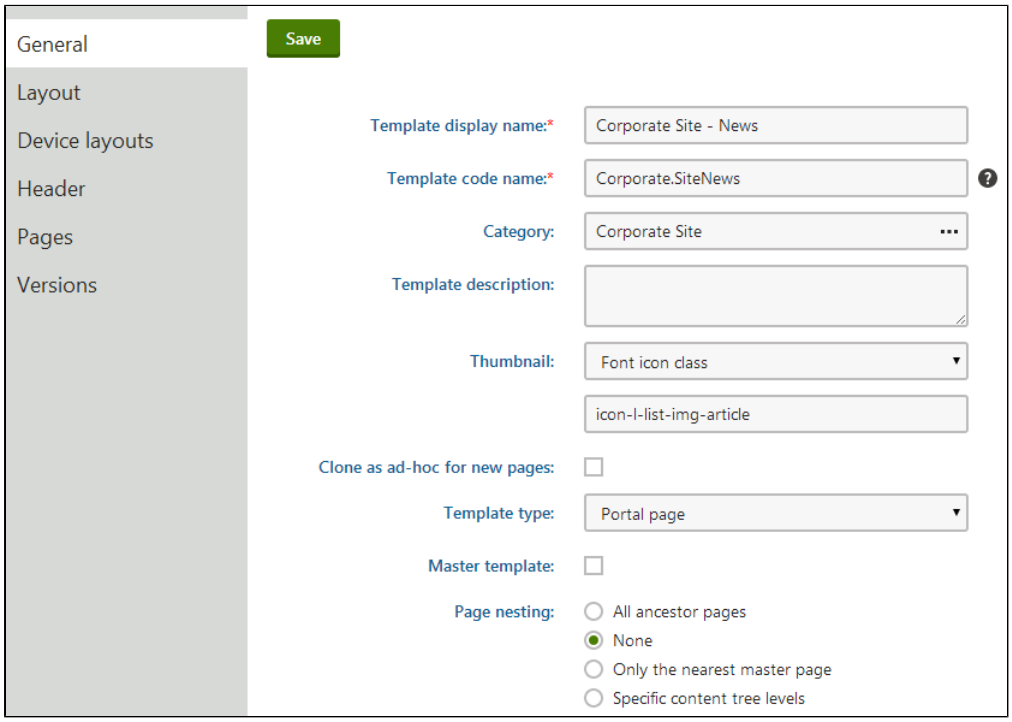
Configuring content inheritance

When using page nesting, you may need to hide parts of the parent page. Kentico provides several ways to achieve this goal:

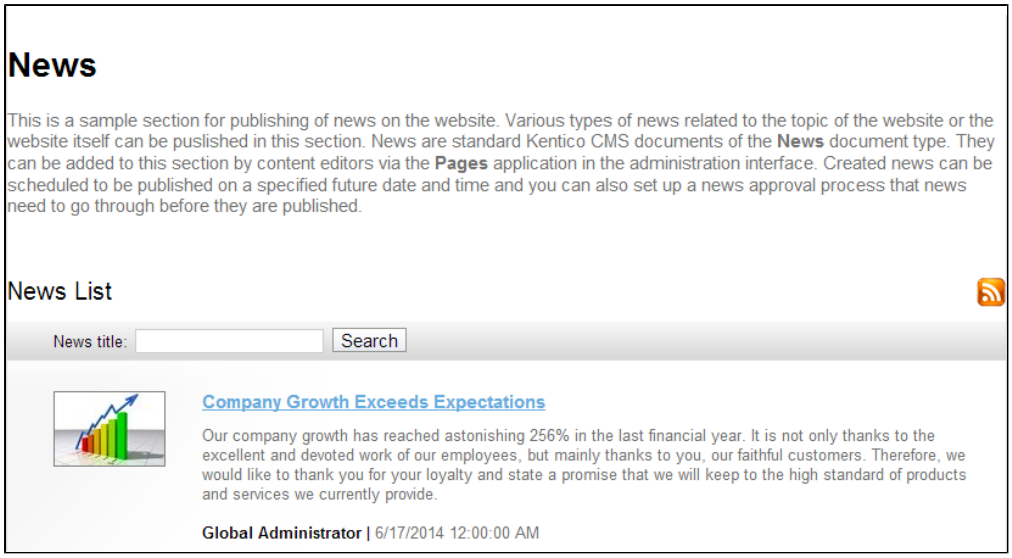
- Using the "Page nesting" settings
- Using the "Hide on subpages" web part property
- Using the "Show for page types" web part property

Using the "Page nesting" settings

1. Open the **Pages** application.
2. Select the **/News** page and open the **Properties -> Template** tab.
3. Click **Edit template properties**.
4. Set the **Page nesting** value to **None**. This means that the page does not use any nesting, so the content from the site's root (master page) is not visible.
5. Click **Save** to confirm the changes.



When you close the dialog and view the News page in **Preview** mode or on the live site, the page does not display the shared content from the master page.



To return the page to its original state, edit the template again and set the **Page nesting** value to **Only the nearest master page**.

You can also set the page nesting for individual pages using the **Properties -> Template** tab. The nesting settings you configure for pages override the settings of the assigned page template.

Using the "Hide on subpages" web part property

Every web part has a property called **Hide on subpages**.

1. In the **Pages** application, edit a parent page on the **Design** tab.
2. Configure (double-click) the web part you want to hide.
3. Expand the **Visibility** category, and enable **Hide on subpages**.
4. Click **Save & Close**.

The web part is hidden on all pages where it is inherited from an ancestor page.

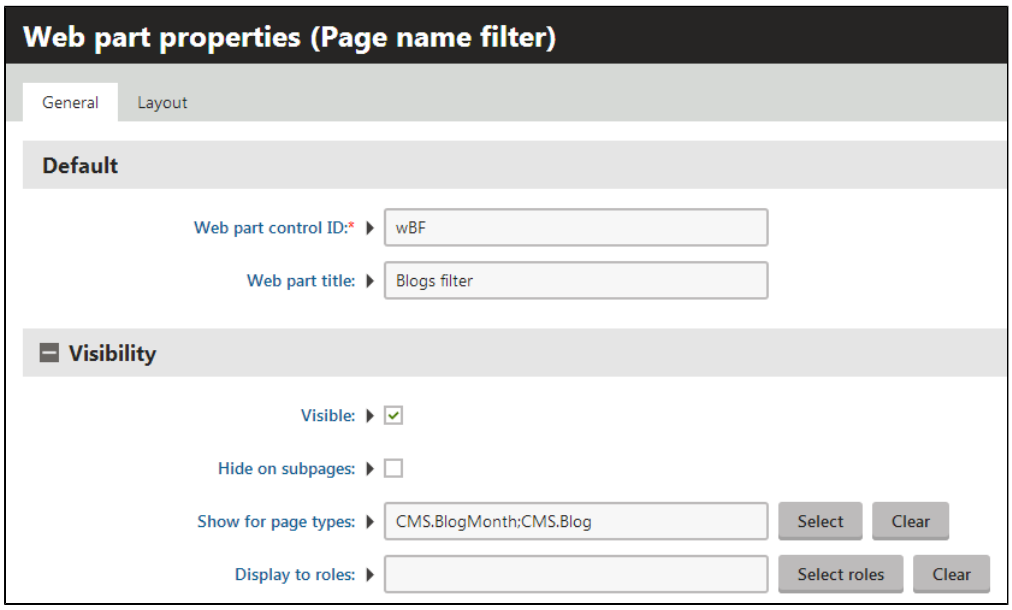
Using the "Show for page types" web part property

The **Show for page types** web part property allows you to limit which page types display the given web part. To see how this works:

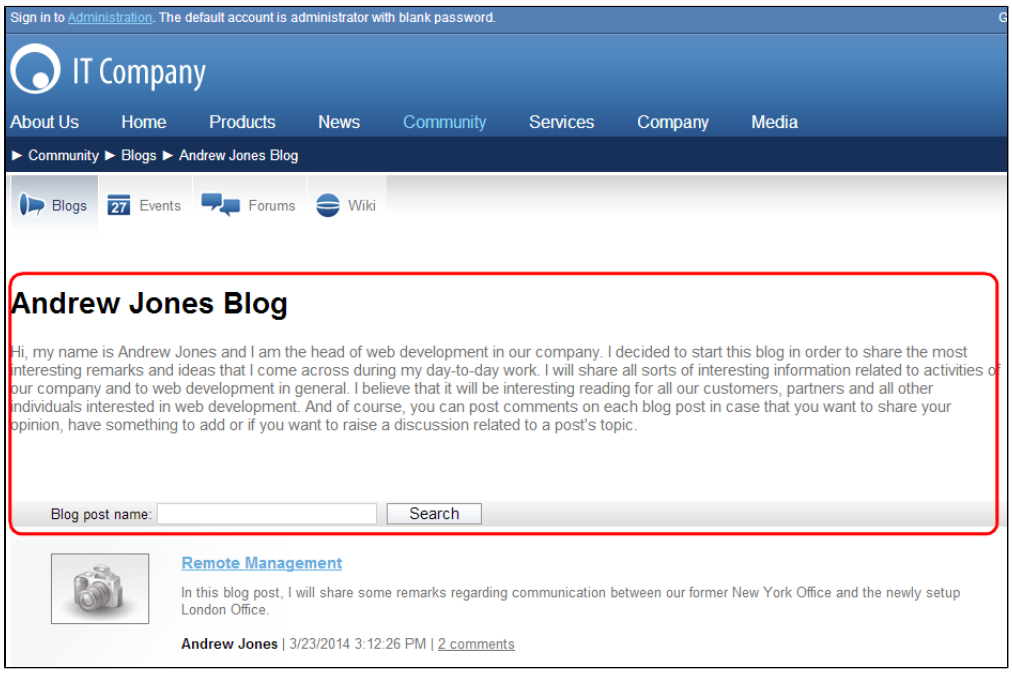
1. In the **Pages** application, select **/Community/Blogs/Andrew Jones Blog** in the content tree.
2. On the **Design** tab, configure (double-click) the **Header text**, **Description text** or **Blogs filter** web part.

   All of the web parts have the **Show for page types** property set to ***CMS.BlogMonth;CMS.Blog*** (in the **Visibility** category). This means that the web parts only appear on blog month and blog pages, not on individual blog posts, which are stored under blog months.

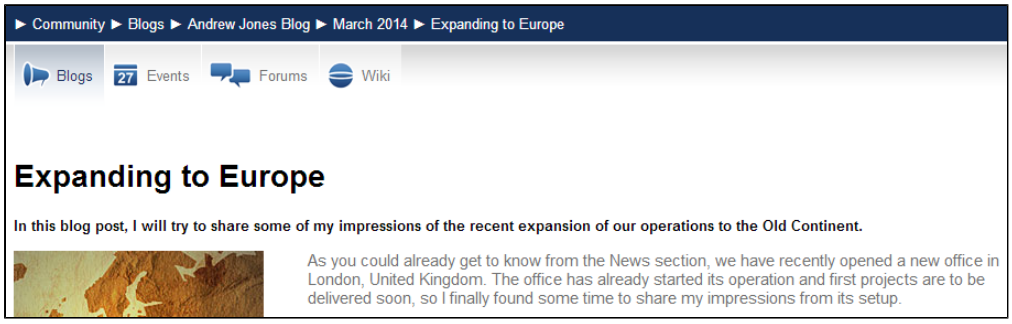
3. Click **Cancel**.



Switch to **Preview** mode. The header text, description text and filter are all displayed above the repeater displaying blog posts.



Click a specific blog post. The web parts are not visible, because the ***CMS.BlogPost*** is not among the allowed page types.



Adding custom code to your website

The easiest way to insert custom code into a portal engine based website is using standard ASCX user controls. If you are not familiar with Visual Studio development, you can skip this page.

Current time example

In this example, we will create a simple user control (ASCX) using Visual Studio and integrate it into the Home page.

1. Open the website project using the **WebSite.sln** or **WebApp.sln** file, which is located in the folder where you installed the website.
2. Create a **Web User Control** and set its name to *GetTime.ascx.*You can set the programming language option to either Visual C# or Visual Basic.
3. Drag the following ASP.NET controls onto the page from the toolbox and set their properties:
    - **Button** control:
        - **ID**: Button1
        - **Text**: Get time
    - **Label** control:
        - **ID**: Label1
        - **Text**: <clear the value>
4. Switch to the **Design** view and double-click the **Show current time** button.
    - The user control's code behind file opens and creates the **Button1_Click** method.
5. Enter the following code into the **Button1_Click** method:

```
Label1.Text = DateTime.Now.ToString();
```

6. Save the user control's files.

The code ensures that the label displays the current date and time when the button is clicked. You do not need to build the project — user controls are compiled at run time.

Adding the user control onto the page

1. Log in to the Kentico administration interface and open the **Pages** application.
2. Select the **Home** page and switch to the **Design** tab.
3. Add a **User control** web part to the **Main zone**.
4. Enter the following value into the web part's **User control virtual path** property*:* ***~/GetTime.ascx***. The ~ character represents the root of your web application.
5. Click **Save & Close**.

Switch to **Preview** mode to see the user control's output on the page. When you click the **Get time** button, the current date and time appears next to the button.

> **User controls versus web parts**
>
> You can also insert custom code onto portal pages by creating your own web parts. Web parts are very similar to user controls, but with a built-in portal engine configuration interface. We recommend building web parts if you need easily re-usable and configurable user controls.
>
> For more information, see Developing web parts in the main documentation.

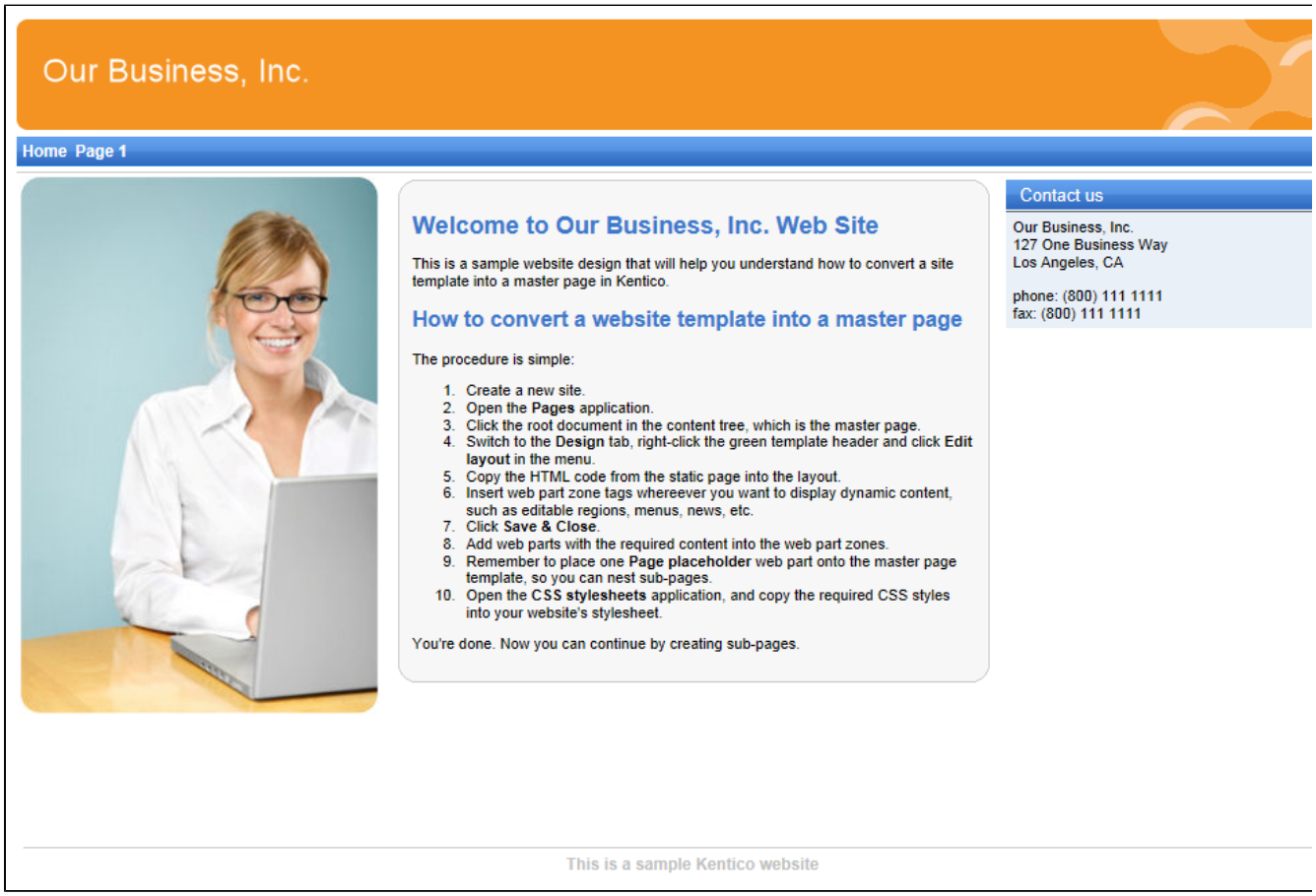
Walkthrough - Creating a new site using the Portal engine

This part of the tutorial guides you through the creation of a simple website. You will learn how to:

- Define site structure and design
- Create your own page templates and pages

The tutorial uses a static website template that is similar to what a developer gets from a graphic designer.

> **Click to download the sample web template**
>
> The template consists of the *home.htm* file, a styles folder and an *app_themes* folder with images.

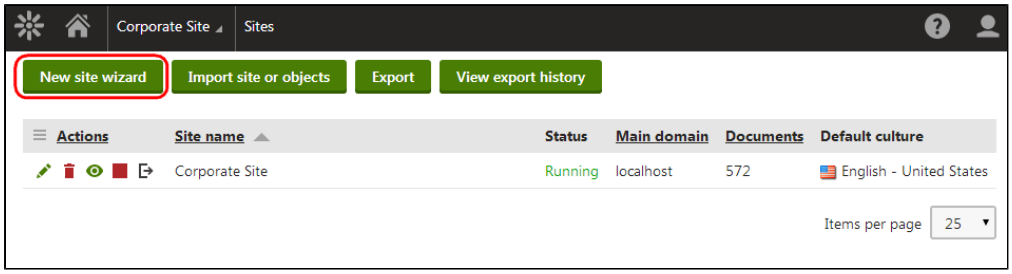


Creating a new website using the New site wizard

The following pages assume that you have previously installed the sample Corporate Site (running on the *localhost* domain). Leave the existing website and add a new site running under the *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 **Corporate Site** site in the **Sites** application. The built-in web server doesn't support any domain other than localhost, so you need to use the ***localhost* domain** ag ain for the new site.

1. Log in to the Kentico administration interface as ***administrator*** and open the **Sites** application.
2. Click **New site wizard**.



The New site wizard opens.

3. Select **Create a new site using a wizard** and click **Next**.

4. Enter the following details for the website:

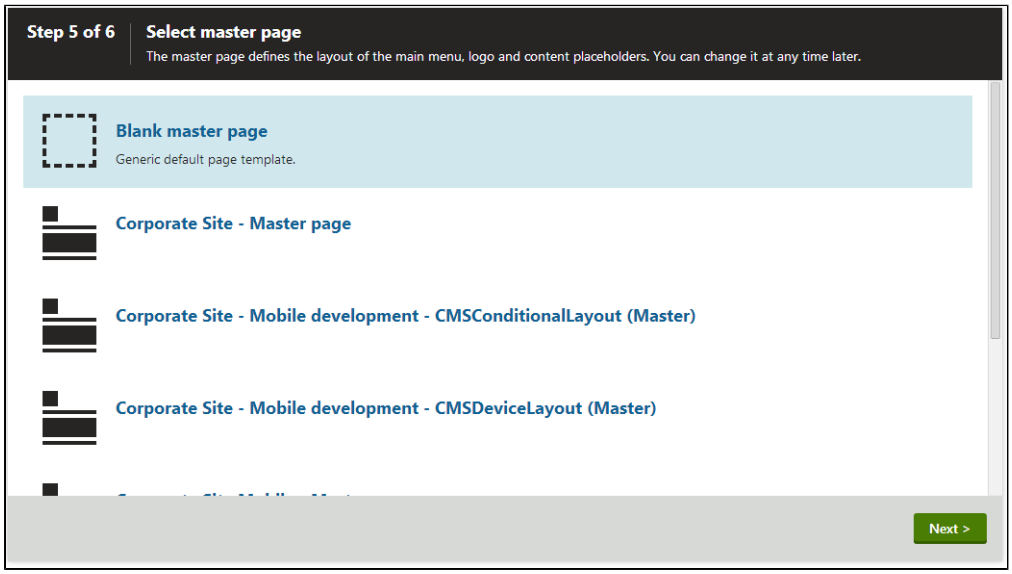
- **Site display name**: My website
- **Site code name**: mysite
- **Domain name**: 127.0.0.1 (if you are using Visual Studio built-in web server, set the Domain name value to *localhost)*
- **Site culture**: English - United States (the default culture determines how the website displays date, time and numeric values based on the different culture-specific format)

Click **Next**.

5. The third 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 fourth step displays the progress of the object import.

6. Choose a master page template for the website. You can change it later at any time. For now, select the **Blank master page**.



Click **Next**. This concludes the initial process of creating the website.

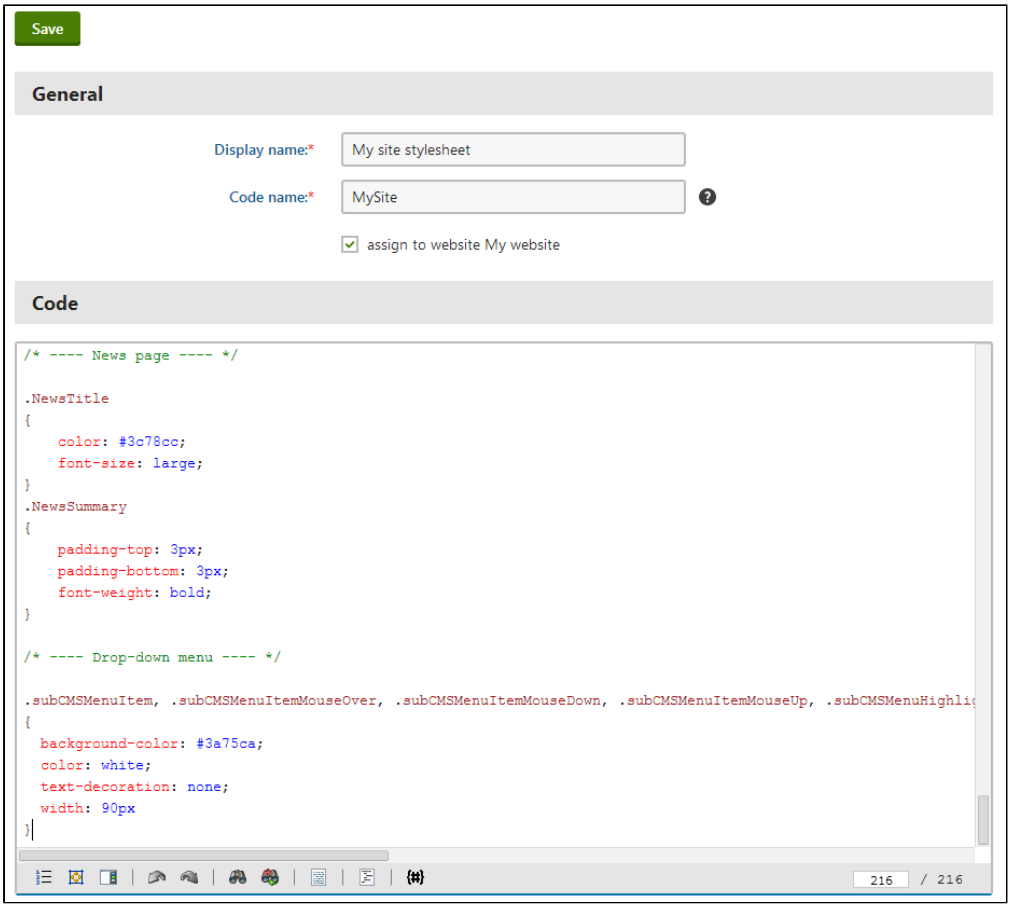
7. Click **Edit your new website**.

The system opens a new tab with the Kentico administration interface using the 127.0.0.1 domain. You need to sign in again (user name **administrator,** blank password) since authentication is not shared over different domains by default.

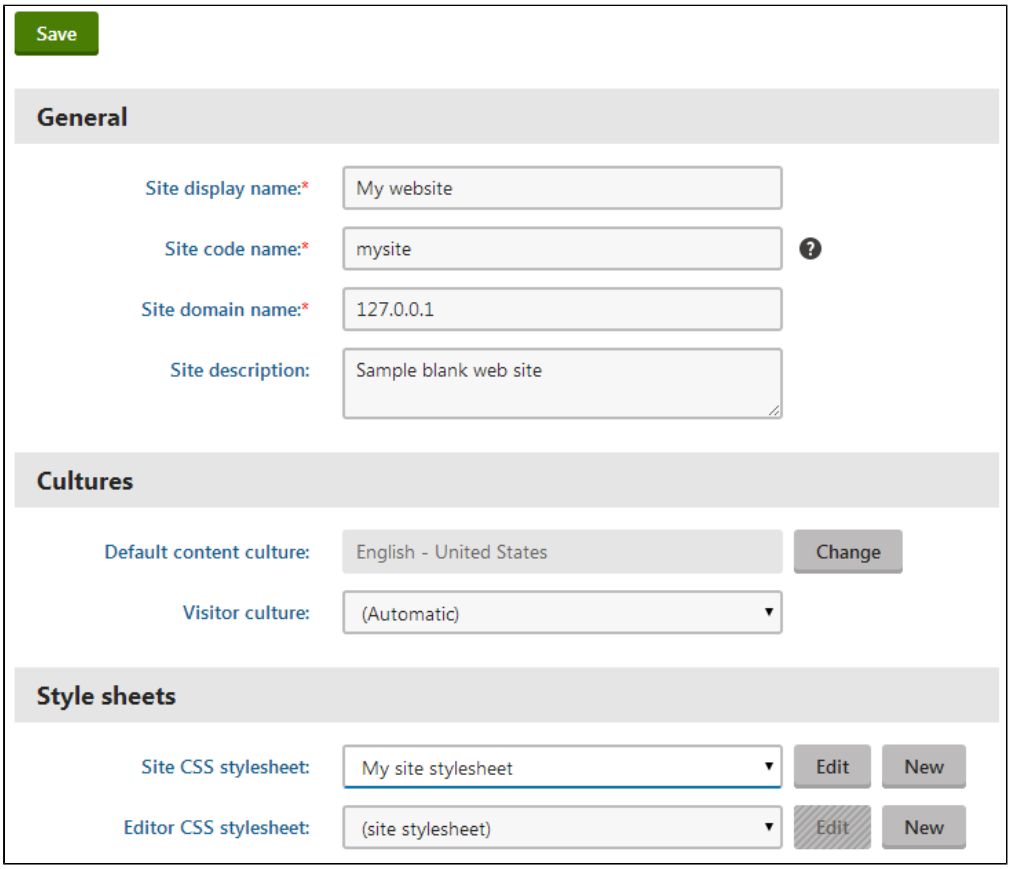
Creating a CSS stylesheet

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

1. Open the **CSS stylesheets** application.
2. Click **New CSS stylesheet**.
3. Enter the following values:
    - **Display name**: My site stylesheet
    - **Code name**: MySite
    - **Code**: copy and paste all CSS code from the Sample web template - **SampleWebTemplate\Styles\main.css**



4. Click **Save**.
5. Switch to the **Sites** tab and assign the stylesheet to **My website**.
6. Open the **Sites** application and edit ( ) **My website**.
7. On the **General** tab, select ***My site stylesheet*** as the **Site CSS stylesheet**.



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 **CMS\App_Themes** folder in your web project (*c:\inetpub\wwwroot\Kentico* by default).

The folder contains graphics for the website template. The App_Themes location ensures that the images are exported as part of the website if you decide to move the website in the future. 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

The image paths in the sample CSS stylesheet already 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

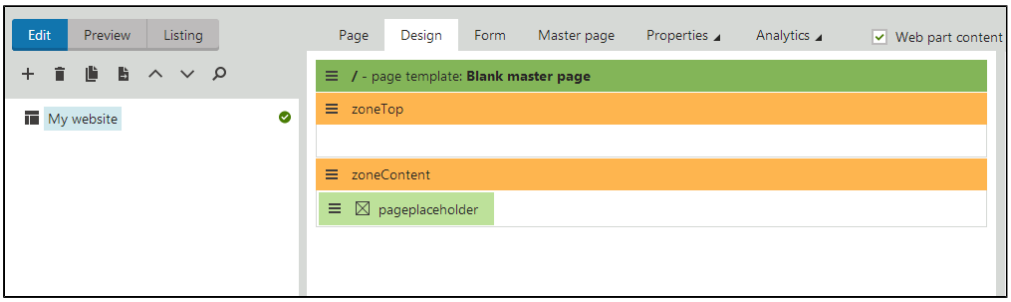
which means that you need to link to files in the App_Themes folder like in the example below:

/app_themes/mysite/images/imagename.gif

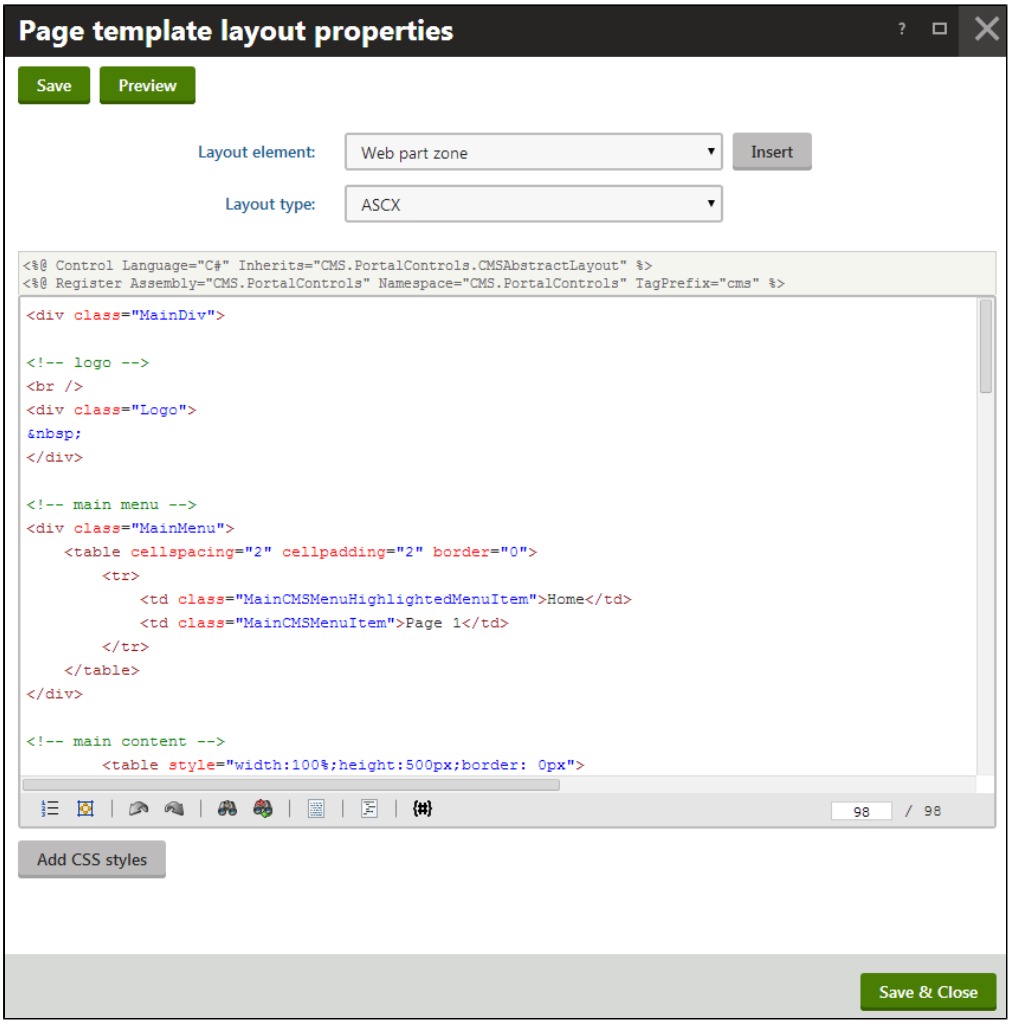
Developing the master page

The master page allows you to define content that can be shared by all pages on the website, such as a header, navigation menu and footer. The site's root page represents the main master page.

1. Open the **Pages** application.
2. Select the root of the content tree (**My website**).
3. Switch to the **Design** tab. The default design uses a blank page template with two web part zones.



4. Right-click the **pageplaceholder** web part in the bottom zone and select **Remove**. The Page placeholder is an important master page component, but you will add it again once the final page layout of the master page is prepared.
5. Click the menu icon ( ) in the green page template header and select **Edit layout** in the menu.
6. Delete the default layout content.
7. Open the sample **home.htm** file (from the Sample web template) and copy the HTML code from inside the **<body>...</body>** tags. Paste this code into the page layout editor.

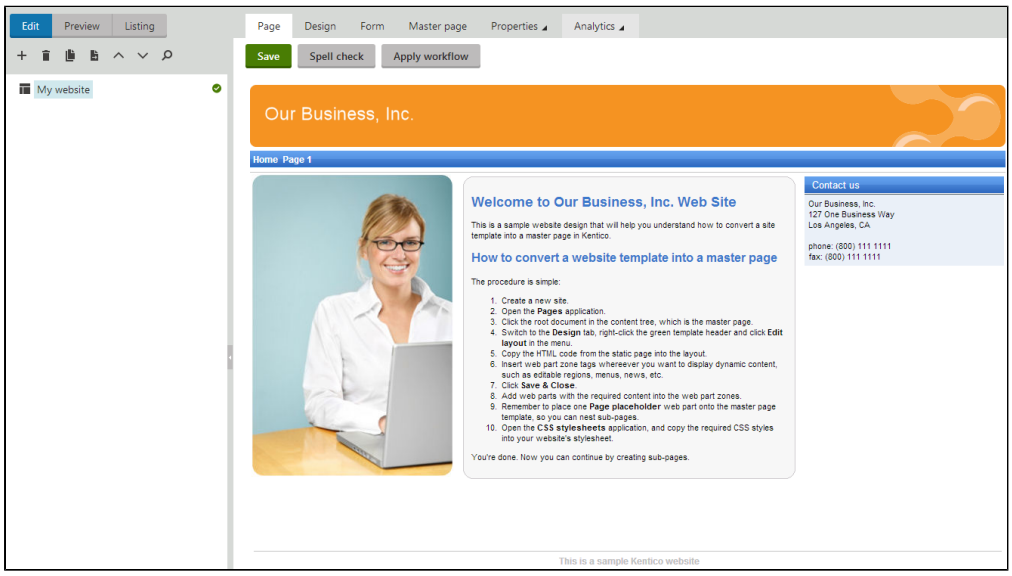


Using CSS-based layout instead of tables

If you prefer using a CSS-based layout, you can modify the HTML code here and replace the tables with other elements (, , etc.).

8. Click **Save & Close**.
9. Switch to the **Page** tab.

Here you can view the new website design. For now, all content is static and determined by the page layout code.



Adding web part zones

To make the master page editable through the portal engine, you need to replace the static HTML code in the page layout with web part zones.

1. In the **Pages** application, edit your site's root page on the **Design** tab.
2. Click the menu icon ( ) in the green page template header and select **Edit layout** in the menu.
3. Locate the **<!-- main menu -->** section in the layout code and delete the entire table inside the ** ** element.
4. Place your cursor inside the ** ** element and click **Insert** above the editor. This adds a web part zone control into the layout.
5. Change the value of the web part zone's **ZoneID** property to ***zoneMenu***. The main menu section should now match the following code:

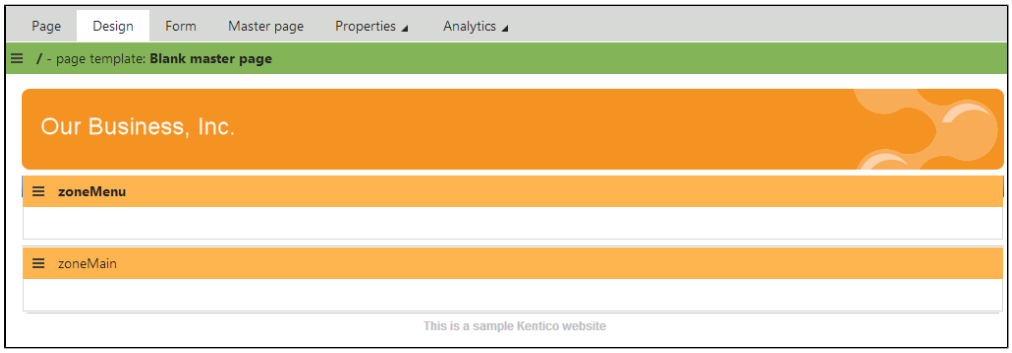
```
<!-- main menu -->
<div class="MainMenu">
    <cms:CMSWebPartZone ZoneID="zoneMenu" runat="server" />
</div>
```

6. Delete the entire content section between the **<!-- main content -->** and **<!-- /main content -->** lines and **Insert** another web part zone instead.
7. Set the second web part zone's **ZoneID** to ***zoneMain***:

```
<!-- main content -->
    <cms:CMSWebPartZone ZoneID="zoneMain" runat="server" />
<!-- /main content -->
```

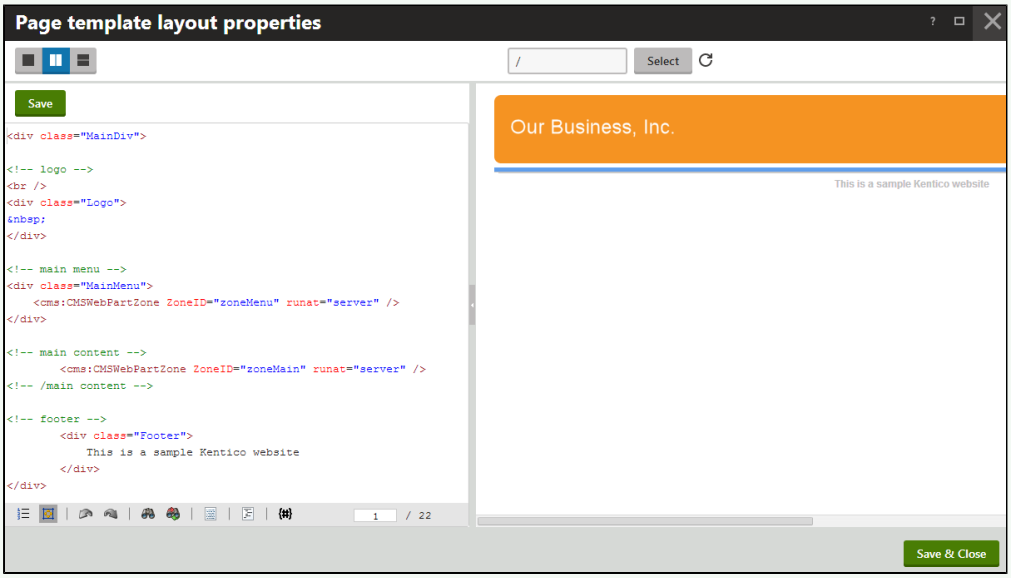
8. Click **Save & Close**.

The **Design** tab now shows the new layout with web part zones.



Previewing the layout

By clicking the **Preview** button in the header of the layout editing dialog, you can modify the code side-by-side with a preview of how the changes affect the live site version of the page.



Master page tab

The **Master page** tab of pages with a master template allows you to add a custom **DOCTYPE** directive, custom **HEAD** section elements and custom **BODY** attributes.

Creating the main menu

Now we will add a dynamic menu to the master page.

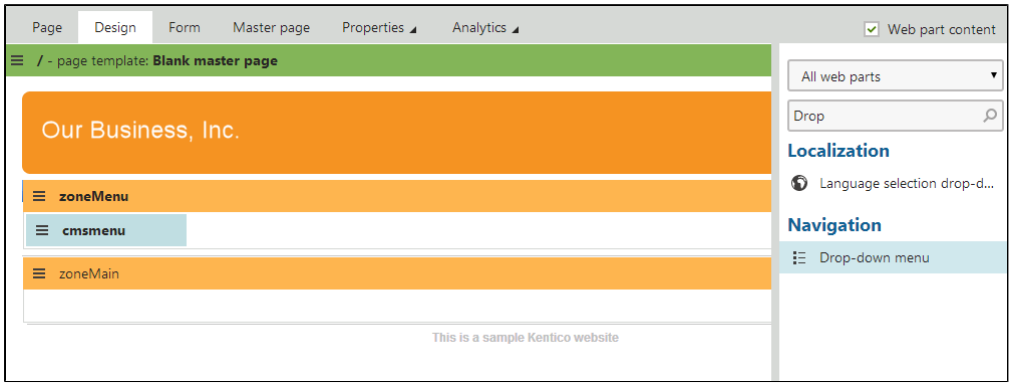
1. In the **Pages** application, select the root page (**My website**).
2. Open the **Design** tab.
3. Add the **Drop-down menu** web part to the **zoneMenu** zone.
4. Set the following properties of the web part (in the **Design** section):

Property	Value	Description
CSSPrefix	;sub	Allows you to add prefixes before the names of the CSS classes applied to the menu. The ;sub value uses unmodified class names for the main (first) menu level and the sub prefix for the second level and all other sub-levels.

Layout	Horizontal	Arranges the menu items horizontally.

5. Click **Save & Close**.

The web part appears in the zone on the **Design** tab.



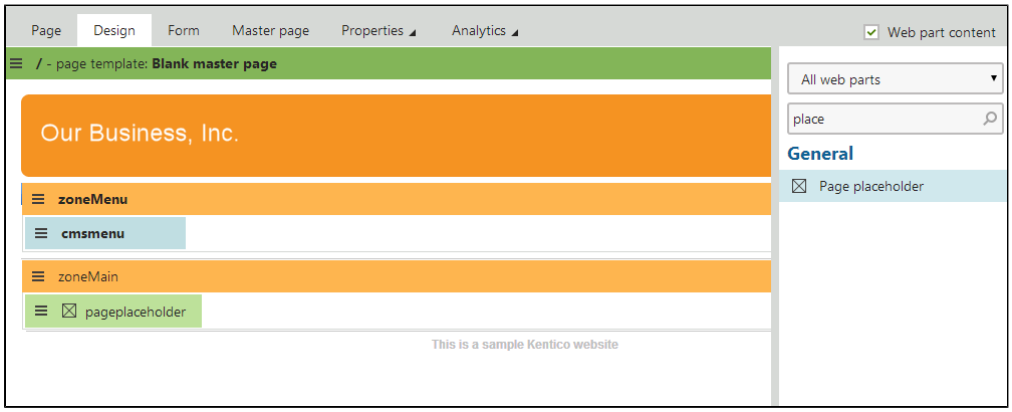
The menu will display the child pages after you create them under the website root.

Adding the page placeholder

Add a **Page placeholder** web part that ensures the loading of child pages (such as Home or News) inside the master template.

1. In the **Pages** application, select the root page (**My website**).
2. Open the **Design** tab.
3. Add the **Page placeholder** web part to **zoneMain**.

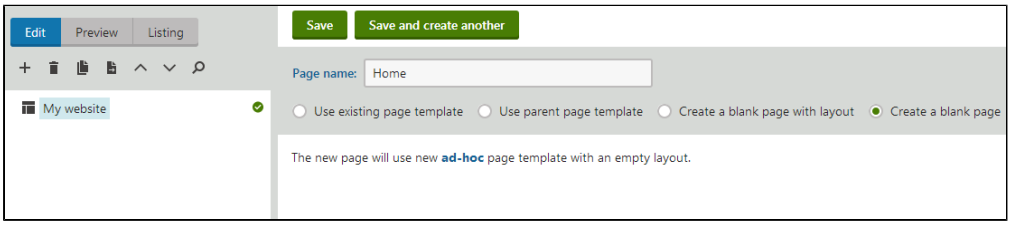
You do not need to make any changes to the page placeholder properties.



Developing the Home page

To create the Home page of the tutorial website.

1. In the **Pages** application, select the root page (**My website**).
2. Click **New** (+).
3. Choose the **Page (menu item)** page type.
4. Type ***Home*** as the **Page name** and choose the **Create a blank page** option.



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

Designing the Home page

1. Switch to the **Design** tab.

2. Click the menu icon ( ) in the green page template header and select **Edit layout** in the menu.
3. Replace the default code with the following:

```
<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;">
              <cms:CMSWebPartZone ZoneID="zoneCenter" runat="server" />
         </td>
            <!-- right column -->
            <td style="padding: 3px 0px 0px 5px;width:270px">
             <cms:CMSWebPartZone ZoneID="zoneRight" runat="server" />
            </td>
        </tr>
</table>
```

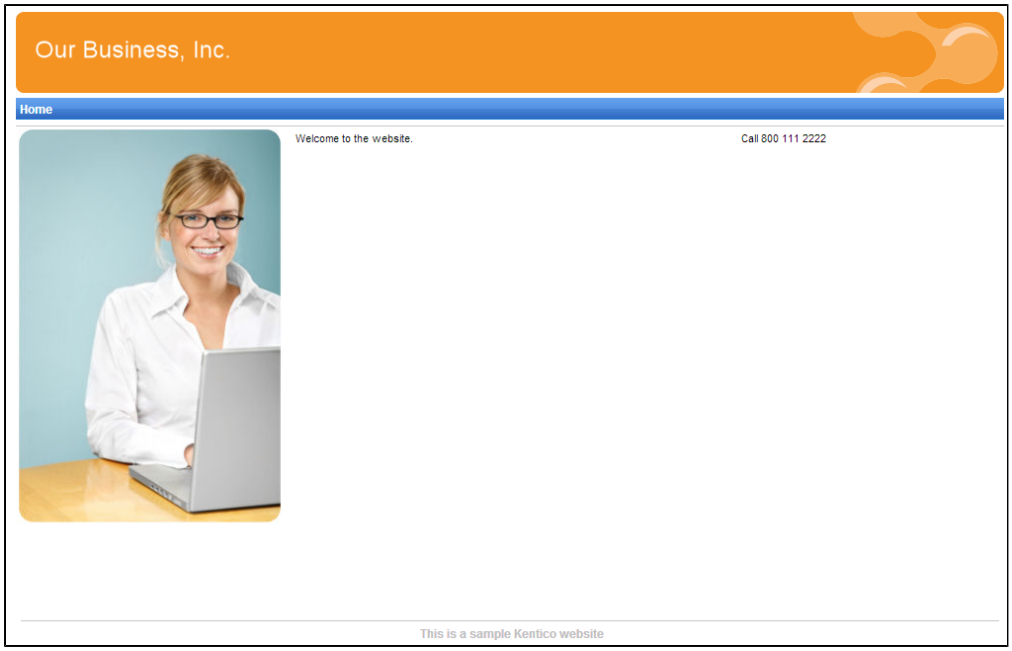
4. Click **Save & Close**.
5. Add the **Editable text** web part into the **zoneCenter** web part zone.
6. Configure (double-click) the web part and set the following properties:
    - **Editable region title**: Main text
    - **Editable region height**: 450

7. Add another **Editable text** web part into the **zoneRight** web part zone. Set its properties:
    - **Editable region title**: Contact text
    - **Editable region height**: 100

8. Open the **Page** tab. The page now contains two editable regions. Enter the following text:
    - Main text: Welcome to the website.
    - Contact text: Call 800 111 2222

9. Click **Save**.

To view the public appearance of your website's new Home page, switch to **Preview** mode.

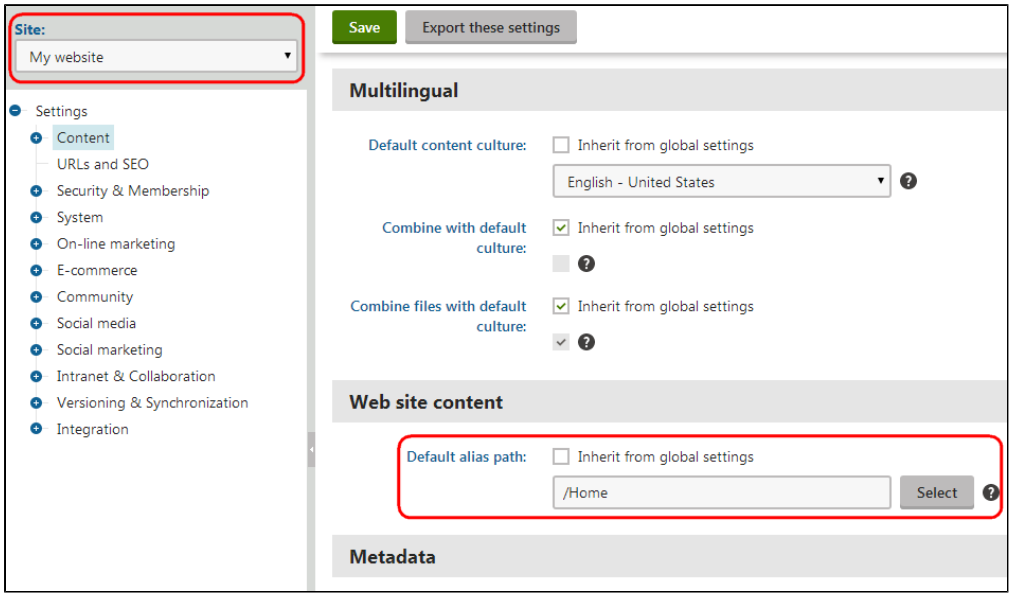


Choosing the website's home page

When a visitor arrives at 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. Open the **Settings** application.
2. Select the **Content** category in the settings tree.
3. Select **My website** in the **Site** drop-down.
4. Clear the **Inherit from global settings** check box next to the **Default alias path** setting.
5. Type in ***/Home***, which is the alias path of your new home page.
6. Click **Save**.



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

Creating a container for the Home page text

Now we will create gray and blue web part containers for our home page text. You can reuse containers for any other web parts later. If you do not wish to use web part containers, you can alternatively insert the surrounding code directly into the HTML layout of your page template.

Creating the web part containers

1. Open the **Web part containers** application.
2. Click **New container**.
3. Enter the following values:
    - **Display name**: My website gray box
    - **HTML code**:

```
<table style="width: 100%;" cellspacing="0" cellpadding="0" border="0"
class="ContainerWithCorners">
 <tr class="ContainerWithCornersRow">
  <td class="ContainerWithCornersTopLeft"> </td>
  <td class="ContainerWithCornersTop"> </td>
  <td class="ContainerWithCornersTopRight"> </td>
 </tr>
 <tr>
  <td class="ContainerWithCornersLeft"> </td>
  <td class="ContainerWithCornersContent" valign="top">

  </td>
  <td class="ContainerWithCornersRight"> </td>
 </tr>
 <tr class="ContainerWithCornersRow">
  <td class="ContainerWithCornersBottomLeft"> </td>
  <td class="ContainerWithCornersBottom"></td>
  <td class="ContainerWithCornersBottomRight"> </td>
 </tr>
</table>
```

The "" character in the code above determines the position of the web part's HTML code within the container.

4. Click **Save**.
5. Switch to the **Sites** tab and assign the container to **My website**.
6. Return to the main container list and create another container with the following values:
    - **Display name**: My website blue box
    - **HTML code**:

```
<table cellpadding="0" cellspacing="0" style="width: 100%;"
class="Blue">
 <tr>
  <td class="BoxTitle">{%ContainerTitle%} </td>
 </tr>
 <tr>
  <td class="BoxArea">

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

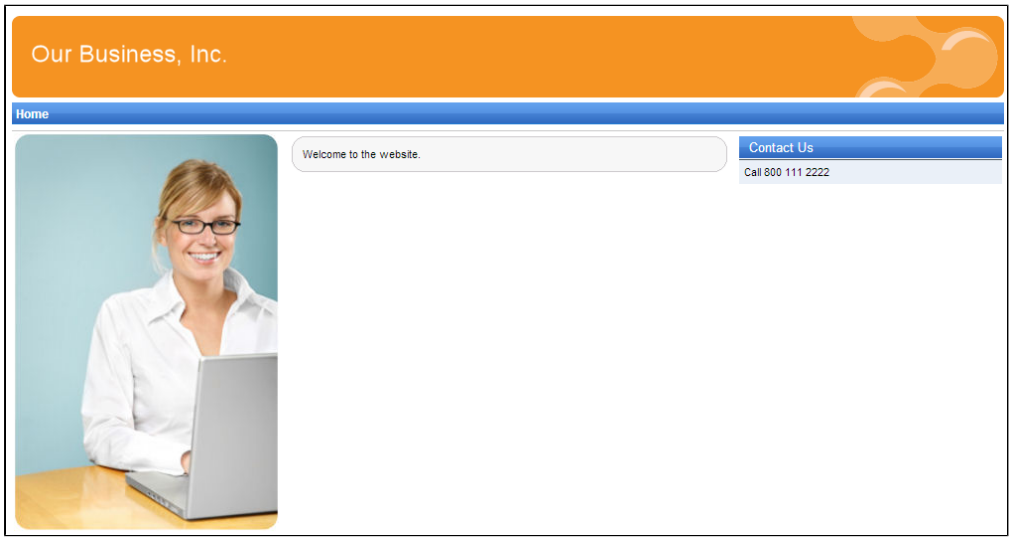
7. Click **Save** and assign the container to **My website** on the **Sites** tab.

Assigning the containers to web parts

Now that the containers are prepared, you can assign them to specific instances of web parts.

1. Open the **Pages** application.
2. Select the **Home** page in the content tree and open the **Design** tab.
3. Configure (double-click) the **editabletext** web part in the **zoneCenter** zone.
4. Scroll down to the **Web part container** property and select ***My website gray box***.
5. Click **Save & Close**.
6. Configure (double-click) the **editabletext1** web part in the **zoneRight** zone
7. Set the following properties in the **Web part container** section:
    - **Web part container**: My website blue box
    - **Container title**: Contact Us
8. Click **Save & Close**.

If you now view the Home page in **Preview** mode, you can see the gray box around the welcome text and the blue box around the contact us text.

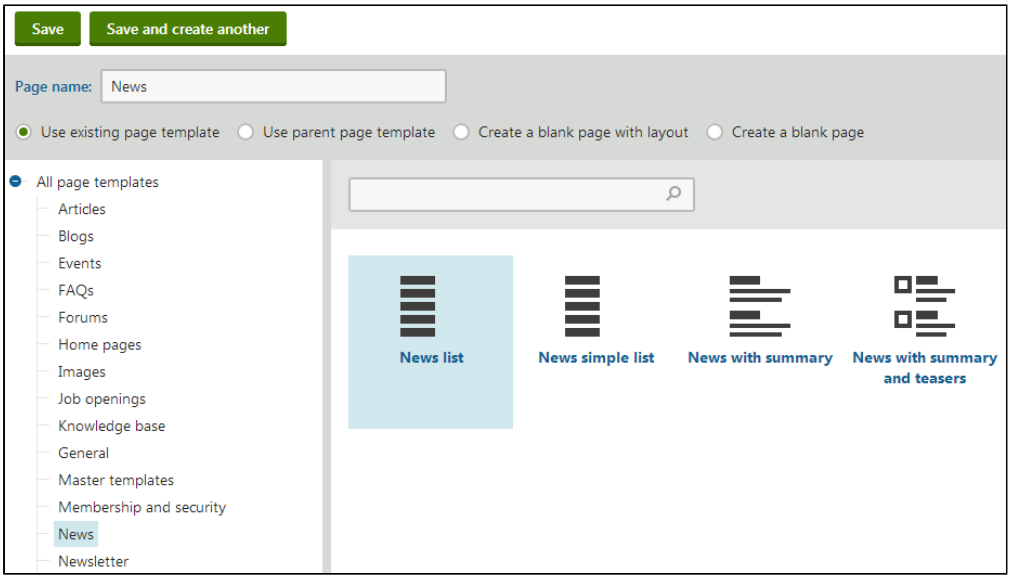


Developing the News page

Now we will create the News section of the tutorial website.

1. In the **Pages** application, select the root page (**My website**).

2. Click **New** (+).
3. Choose the **Page (menu item)** page type.
4. Type ***News*** as the **Page name** and select the **News -> News list** page template.



5. Click **Save**.

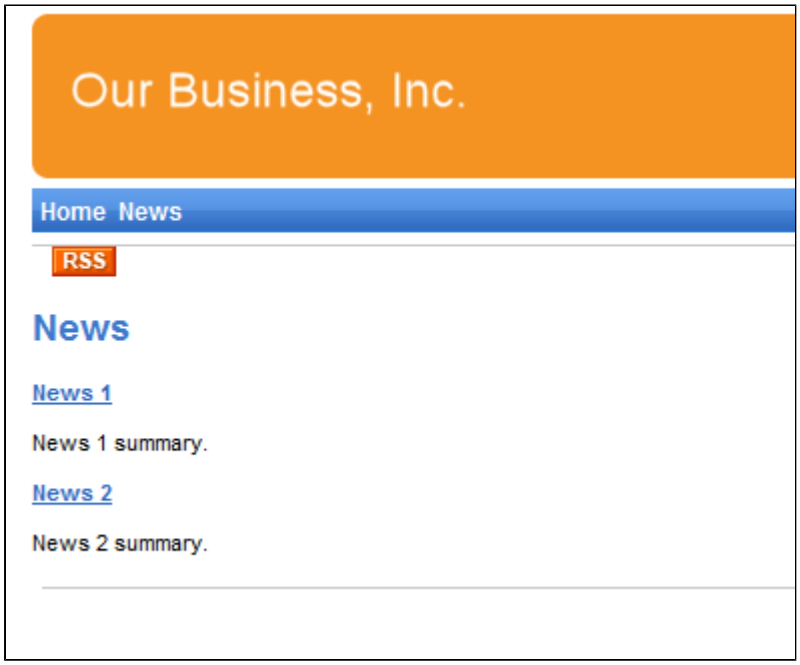
The News page uses a predefined template, so it is ready without any further modifications.

Creating news articles

To add news articles to the website:

1. In the **Pages** application, select the **/News** page.
2. Click **New** (+).
3. Choose the **News** page type.
4. Fill in the news page fields with the following values:
    - **News Title**: News 1
    - **Release Date**: click **Today**
    - **News Summary**: News 1 summary.
    - **News Text**: News 1 text.
    - **Publish from/Publish to**: leave the fields blank
5. Click **Save and create another** and enter the following values:
    - **News Title**: News 2
    - **Release Date**: click **Today**
    - **News Summary**: News 2 summary.
    - **News Text**: News 2 text.
    - **Publish from/Publish to**: leave the fields blank
6. Click **Save**.

If you select the /News page and switch to **Preview** mode, you can see a list of all news pages placed under the **News** section.



This is an example of how content is structured in Kentico. 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 accessible to both people and search engines.

How does the News list work?

You may be wondering how Kentico generates the news list. The news page is a good example of using web parts to display structured page data on the website.

Select the **News** page, switch to **Edit** mode and open the **Design** tab. The page contains several web parts, including the **NewsRepeater**. This web part is based on the Repeater web part, which loads content from the database. Configure (double-click) the web part to view how the web part's properties are set.

The most important properties are the following:

Property	Value	Description
Path		Determines the path of the pages in the content tree from which the web part loads and displays data. If you leave the value empty, the web part displays all pages under the current path (for pages) or displays the details of the selected page (for News pages). See Path expressions
Page types	cms.news	This property determines what types of pages the web part displays (news, products, blogs etc.).

Transformation	cms.news.preview	Assigns the transformation that the web part uses to display the list of news items. A transformation is a piece of code which converts raw database data of a page into user-friendly HTML output.
Selected item transformation	cms.news.default	When a user selects a specific news item on the website, the repeater displays the details according to the specified transformation.

The following steps show how the page handles visitors:

1. A visitor arrives on the **/News** page.
2. The **Repeater** web part placed on the page template checks if a news page is currently selected (based on the value of the **Page types** property).
3. The web part finds out that the current page is a **Page (menu item)**, so it looks for all underlying news pages 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 web part uses the **cms.news.default** transfor mation instead to display the details.

Path expressions

Listing web parts and controls have the **Path** property that specifies which content the component loads and displays. The following expressions are examples that you can use to select pages:

Path expression	Meaning
/%	All pages on the website.
/news/%	All pages under /News.
/news/news1	The News1 page.
./%	All items under the current page.
./logo	The Logo page under the current page.
./images/%	All pages under the Images page, which is a child of the current page.
../contacts/%	All pages under the Contacts page on the same content level as the current page.
/{0}/%	All pages under the page located on the first level of the current path. Example: If the currently selected page is: */news/news1* the system evaluates the expression as: */news/%*

Developing the Services page

Now we will add a website section displaying information about services. The page template used for this section will contain a tree menu on the left and two editable regions.

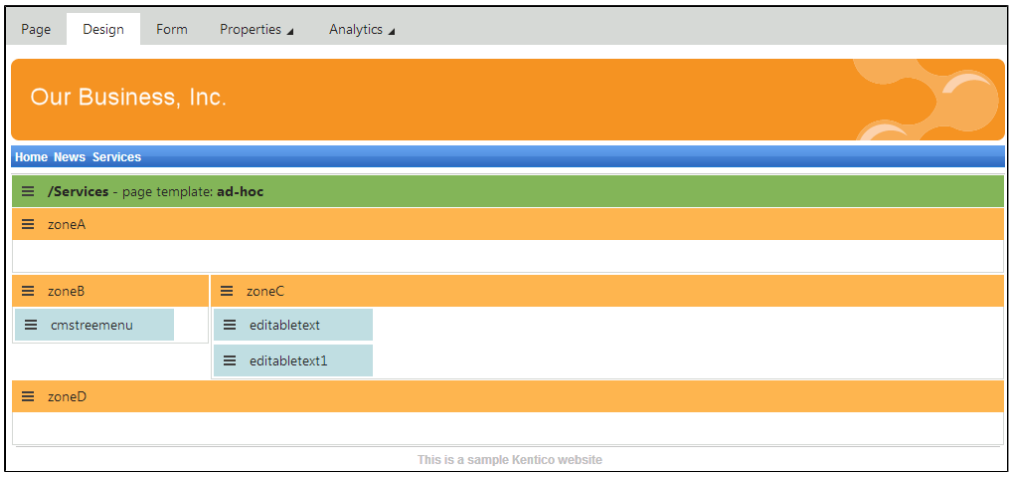
1. In the **Pages** application, select the root page (**My website**).
2. Click **New** (+).
3. Choose the **Page (menu item)** page type.
4. Type ***Services*** as the **Page name** and choose the **Create a blank page with layout** option.
5. Select the **Two columns - 20/80** layout with the **Copy this layout to my page template** box checked.
6. Click **Save** to create the page.
7. Open the **Design** tab and add the **Tree menu** web part into the **zoneB** zone (left zone on the middle row).
8. Set the following properties for the **Tree menu** web part:

Property	Value	Description
Content -> Path	/{0}/%	Configures the tree menu to display pages starting from the second level of the currently selected path.
Design -> Item image URL	~/app_themes/mysite/images/bullet.gif	Sets the path of the image that the menu displays next to items. The ~ character represents the root of the website. This relative path ensures that the web part displays the images correctly even if the website's virtual directory name changes.

Design -> Open item image URL	~/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 page.

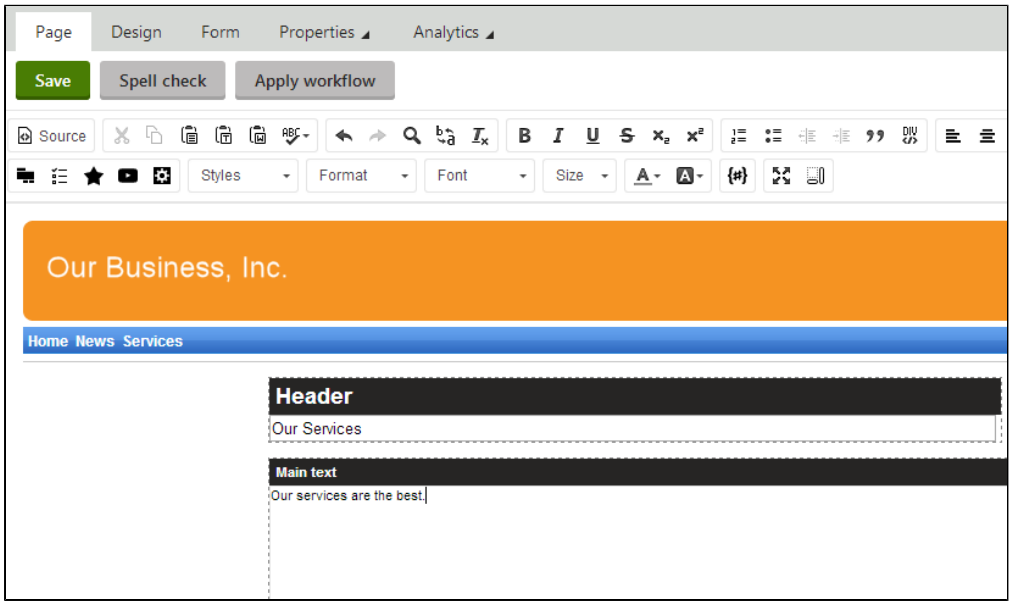
9. Click **Save & Close**.
10. Add the **Editable text** web part into the **zoneC** zone (right zone on the middle row) and set the following properties:
    - **Editable region title**: Header
    - **Editable region type**: Text box
    - **Editable region width**: 600
    - **Content before**: <h1>
    - **Content after**: </h1>
11. Add another **Editable text** web part below the first one and set the following properties:
    - **Editable region title**: Main text
    - **Editable region type**: HTML Editor
    - **Editable region height**: 300

The **Design** tab of the Services page should now look like this:



Entering text

1. Switch to the **Page** tab of the **Services** page.
2. Type some text into the editable regions.
3. Click **Save.**



Creating a re-usable page template

Save the current state of the page as a re-usable page template.

1. With the **Services** page selected, go to the **Properties -> Template** tab.
2. Click **Save as new template**.
3. Fill in the following values for the new template:
    - **Template display name**: Services template
    - **Template category**: General
    - **Assign to the current page**: yes (checked)
4. Click **Save & Close**.

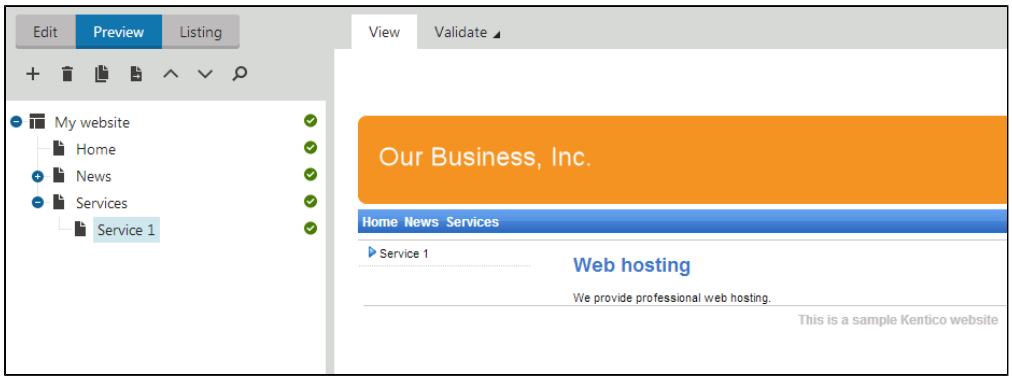
You can now select the **Services** template directly when creating new pages.

Adding sub-pages

Now use the Services template to add separate pages containing information about individual services under the Services page.

1. In the **Pages** application, select the **/Services** page in the content tree.
2. Click **New** (+).
3. Choose the **Page (menu item)** page type.
4. Type in **Service 1** as the **Page name** and choose the **Use existing page template** option.
5. Select the **General** category and the **Services template** page template.
6. Click **Save**.
7. Type the following text into the editable regions on the **Page** tab:
    - **Header**: Web hosting
    - **Main text**: We provide professional web hosting.
8. Click **Save**.

To see how the Service 1 page appears to website visitors, switch to **Preview** mode.



The Services section demonstrates how to create a new page template from scratch and re-use it for any number of pages. The system stores the text of editable regions separately for every page, so you can enter unique content on each page.

Developing the Products section

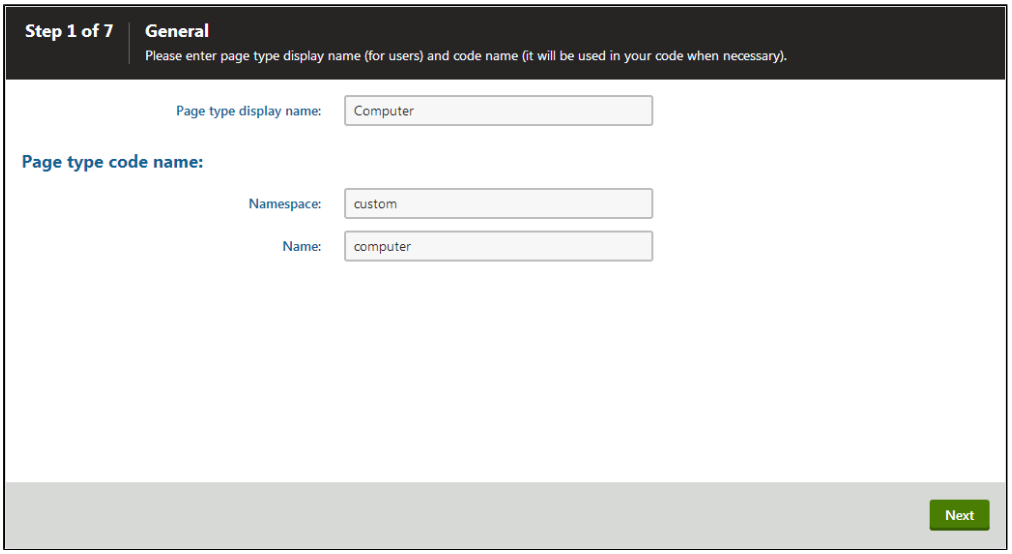
Now we will add a products section displaying a list of computers and their technical specifications. You will learn how to:

1. Create a new page type representing computer products
2. Write transformations
3. Create a page displaying a list of computer products

Defining a new page type

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

1. Open the **Page types** application.
2. Click **New page type**. This starts the **New page type** wizard.
3. Enter the following values in **Step 1**:
    - **Page type display name**: *Computer* (the system displays this name to users in the administration interface)
    - **Namespace**: *custom* (namespace to distinguish your page types from the default system types that use the cms namespace)
    - **Name**: *computer* (the identifier of the page type)



4. Click **Next**.
5. In **Step 2**, specify the name of the database table where the system stores the data of computer pages. You also need to enter the name of the table's primary key field. Leave the default values.
6. Click **Next**. The system creates a new database table for computer pages.
7. In **Step 3**, you need to define the fields of the page type (columns of the table). Click **New field** to create the following fields. For each field, enter the values, click **Save** and repeat the procedure until you have defined all the listed fields.

    - **Field name**: ComputerName
    - **Data type**: Text
    - **Size**: 200
    - **Required:** yes (checked)
    - **Field caption**: Computer name
    - **Form control**: Text box

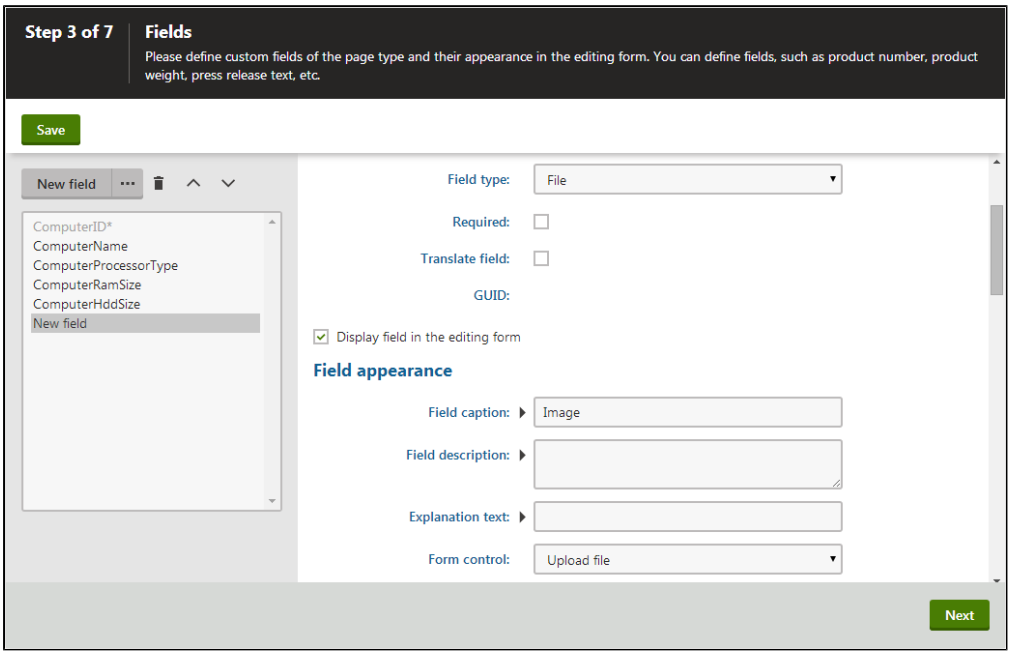
    - **Field name**: ComputerProcessorType
    - **Data type**: Text
    - **Size**: 200
    - **Field caption**: Processor type
    - **Form control**: Drop-down list
    - **Editing control settings -> Data source**: select **List of options** and enter the following items into the text area, one per line:

```
Athlon;Athlon
Pentium XEON;Pentium XEON
Pentium Core 2 Duo;Pentium Core 2 Duo
```

    - **Field name**: ComputerRamSize
    - **Data type**: Integer number
    - **Field caption**: RAM (MB)
    - **Form control**: Text box

    - **Field name**: ComputerHddSize
    - **Data type**: Integer number
    - **Field caption**: HDD (GB)
    - **Form control**: Text box

    - **Field name**: ComputerImage
    - **Data type**: File
    - **Field caption**: Image
    - **Form control**: Upload file



8. Click **Next**.
9. In **Step 4**, choose the **ComputerName** field as the **Page name source**.

> This means that when a user creates a new computer page, the system automatically fills in the page name based on the **ComputerName** value. The page name appears in site navigation and in the content tree of the **Pages** application.

10. Click **Next**.
11. In **Step 5**, select the page types that will be supported as parents for computer pages in the content tree. Click **Add page types**, select the **Page (menu item)** page type and click **Select**. This means that users are only allowed to place computer pages under pages, not under articles, news items or other page types.
12. Click **Next**.
13. In **Step 6**, assign the page type to all websites where you wish to use it. Click **Add sites**, choose **My website** in the selection dialog and click **Select**.
14. Click **Next**.
15. Click **Finish** to complete the creation of the new page type.

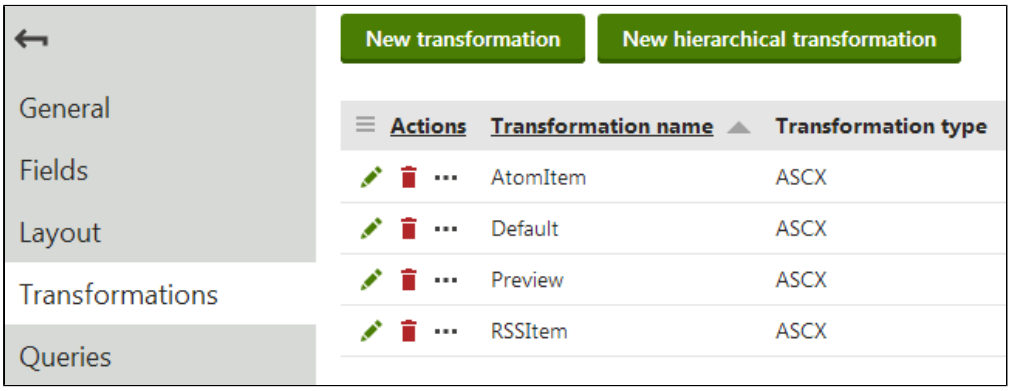
The wizard automatically creates the database table and several default transformations.

> **How does the system store page content?**
>
> The system stores page content and all related data in three database tables:
>
> - **CMS_Tree** (content tree structure)
> - **CMS_Document** (general page properties, metadata and editable region content)
> - A **dedicated page type table** - in this case ***CUSTOM_Computer*** (stores the values of the page type's specific fields)

Writing transformations

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

1. Open the **Page types** application.
2. Edit ( ) the **Computer** page type.
3. Switch to the **Transformations** tab.



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

4. 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>
```

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

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

5. Click **Save**.
6. 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>
```

7. Click **Save.**

Note the code used to create the link to specific pages. 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 page's **ComputerImage** field using the **GetImage** method. The sample code calls the method with a parameter that ensures automatic serverside 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 pages 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 press CTRL + SPACE to access a list of available page fields and related objects instead of typing them manually.

Creating the Products page

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

Creating the product list page

1. Open the **Pages** application.
2. Select the root page (**My website**).
3. Click **New** (+).
4. Choose the **Page (menu item)** page type.
5. Type ***Products*** as the **Page name** and choose the **Create a blank page** option.
6. Click **Save** to create the page.
7. Switch to the **Design** tab and add the **Breadcrumbs** web part into **zoneA**. Leave the default properties for the web part and click **Save & Close**.
8. Add the **Datalist** web part below the breadcrumbs. Set the following properties for the web part:

Property	Value	Description
Page types	custom.computer	Configures the datalist to display only pages of the custom.computer type (created in the Defining a new page type topic).
ORDER BY expression	ComputerName ASC	Sets the SQL ORDER BY clause that the web part uses when loading data. As a result, the datalist displays items in ascending alphabetical order based on the values of the ComputerName field.
Transformation	custom.computer.preview	Assigns the transformation that the datalist uses to display the list of computer products.
Selected item transformation	custom.computer.default	When a user selects a specific computer page on the website, the web part displays the details according to the specified transformation.

9. Click **Save & Close**.

The page is now ready to display underlying computer pages.

Adding computer pages

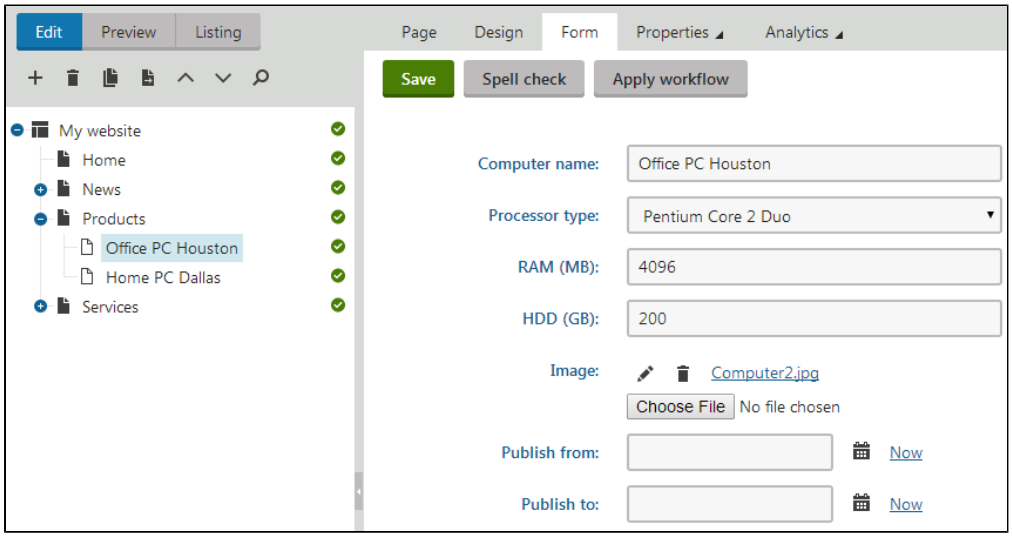
Now add pages representing individual computer products:

1. In the **Pages** application, select the **Products** page in the content tree.
2. Click **New** (+) and choose the **Computer** page type.
   - **Computer name**: Home PC Dallas
   - **Processor type**: Athlon
   - **RAM (MB)**: 2048
   - **HDD (GB)**: 160
   - **Image**: upload an image (you can find images in the Sample web template - **SampleWebTemplate\Computer_Images**)
   - **Publish from/Publish to**: leave the values blank

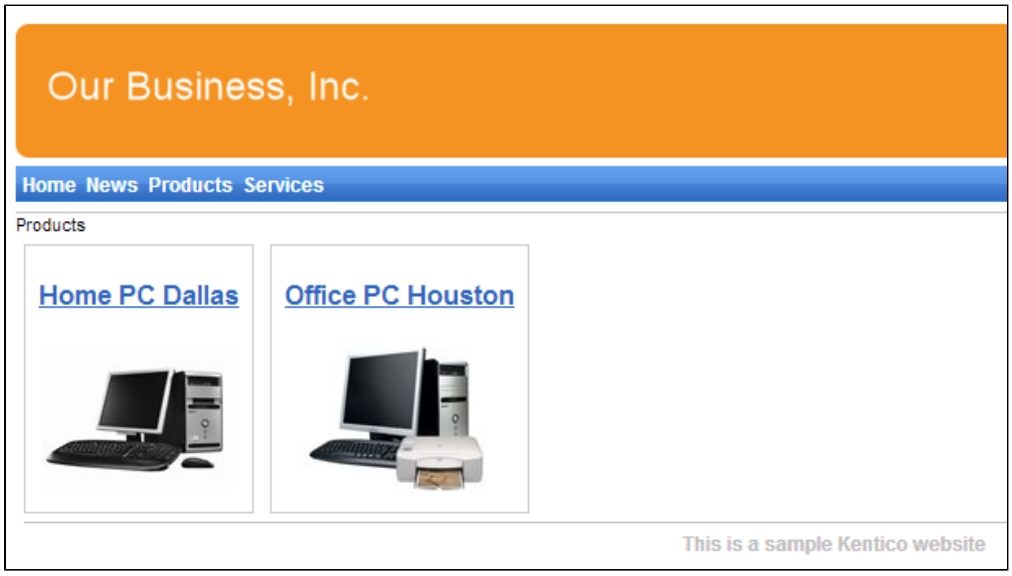
3. Click **Save and create another** and enter the following values:
    - **Computer name**: Office PC Houston
    - **Processor type**: Pentium Core 2 Duo
    - **RAM (MB)**: 4096
    - **HDD (GB)**: 200
    - **Image**: upload an image (you can find images in the Sample web template - **SampleWebTemplate\Computer_Images**)
    - **Publish from/Publish to**: leave the values blank

4. Click **Save**.

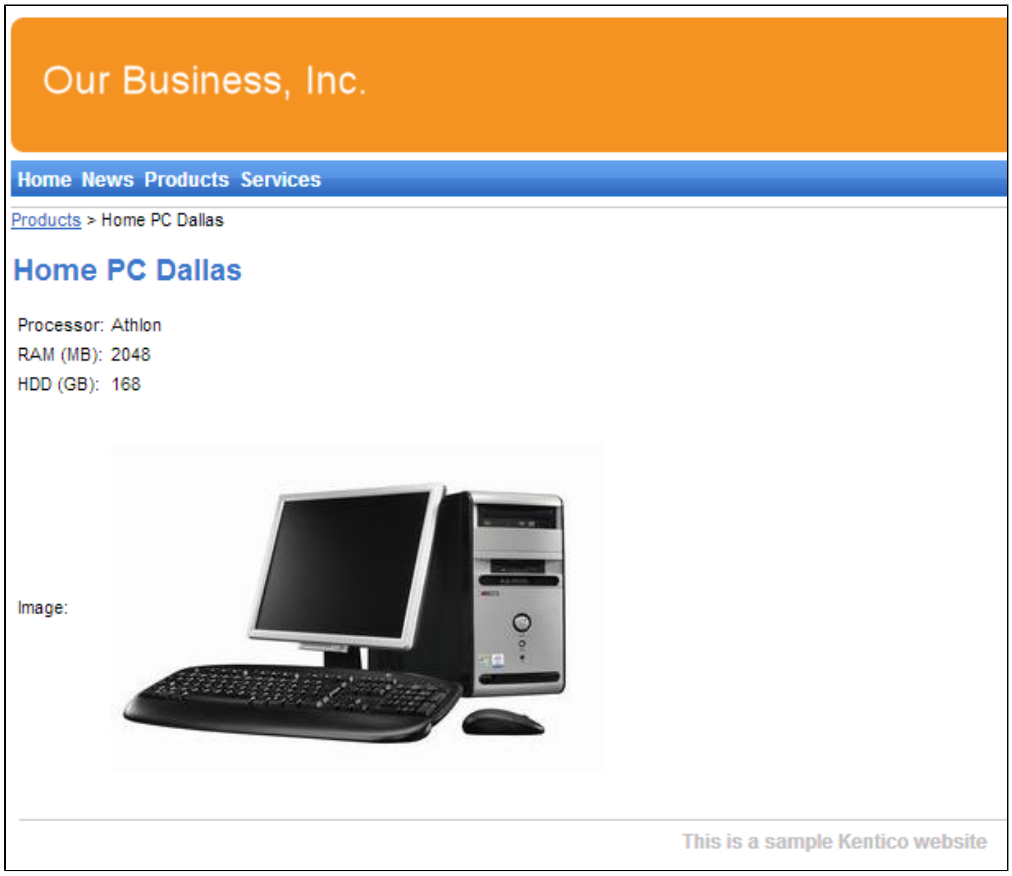
You can edit the field values of the computer pages at any time by switching to the **Form** tab in the **Pages** application.



If you view the /Products page in **Preview** 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).



Developing the Search page

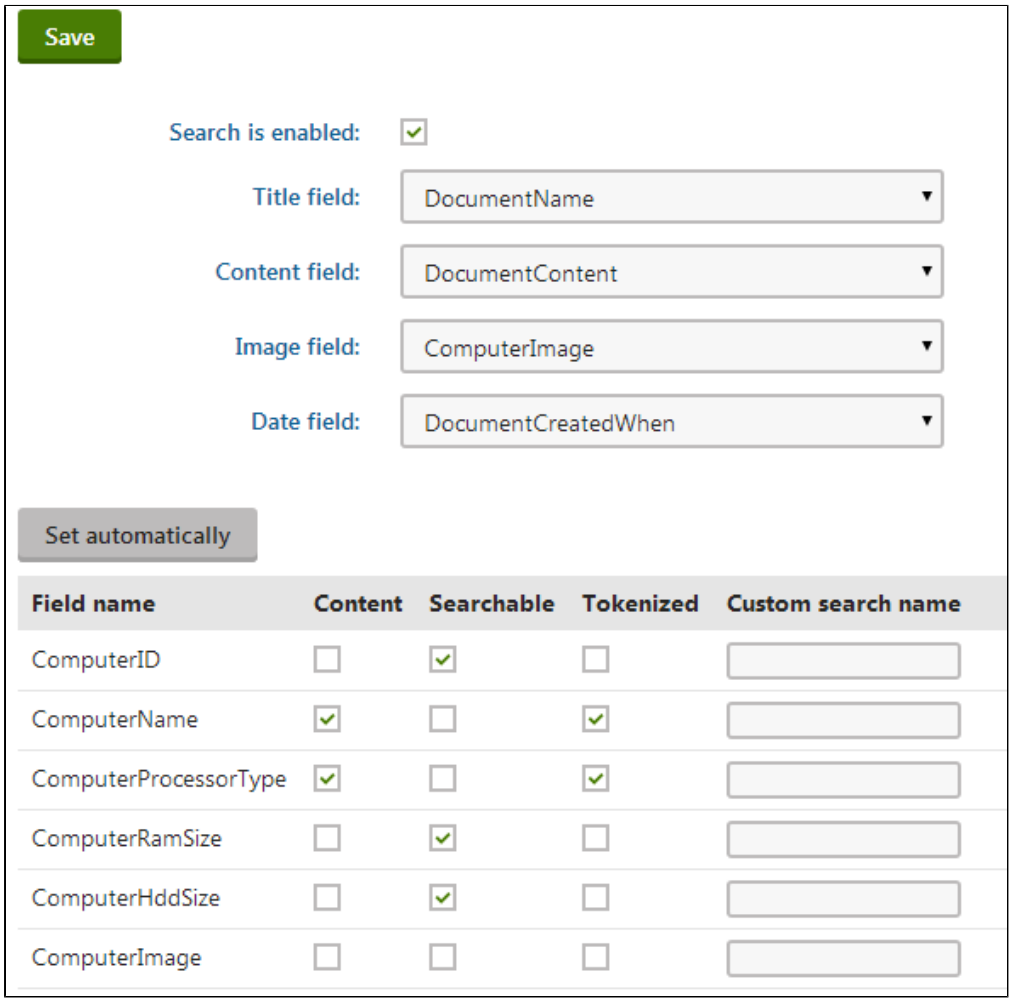
Kentico allows users to perform index-based searches through all page content, as well as other types of data. The following instructions describe how to add a basic search page to your website.

Configuring search fields for the Computer page type

First, set up the search options for the **Computer** page type that you created for the Products section.

1. Open the **Page types** application.
2. Edit ( ) the **Computer** page type.
3. Switch to the **Search fields** tab.
4. Select the **Search is enabled** check box.
5. Set the **Image field** to ***ComputerImage.***
6. Click **Save.**

Product pages are now searchable.

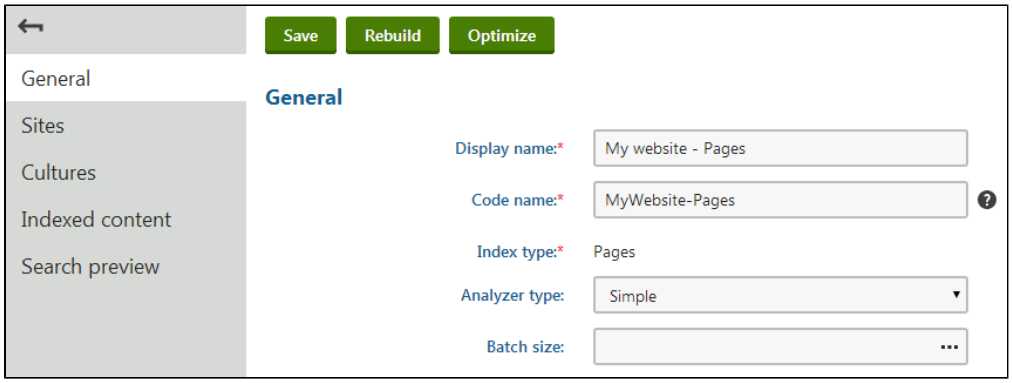


Creating a smart search index

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

1. Open the **Smart search** application.
2. Click **New index**.
3. Fill in the following details for the search index:
    - **Display name**: My website - Pages
    - **Index type**: Pages
    - **Analyzer type**: Standard
    - **Stop words**: (default)
4. Click **Save**. The index's editing interface opens.
5. Open the **Indexed content** tab and click **Add allowed content**.
6. Type **/%** into the **Path** field and click **Save**. This ensures that the index includes all pages 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**.

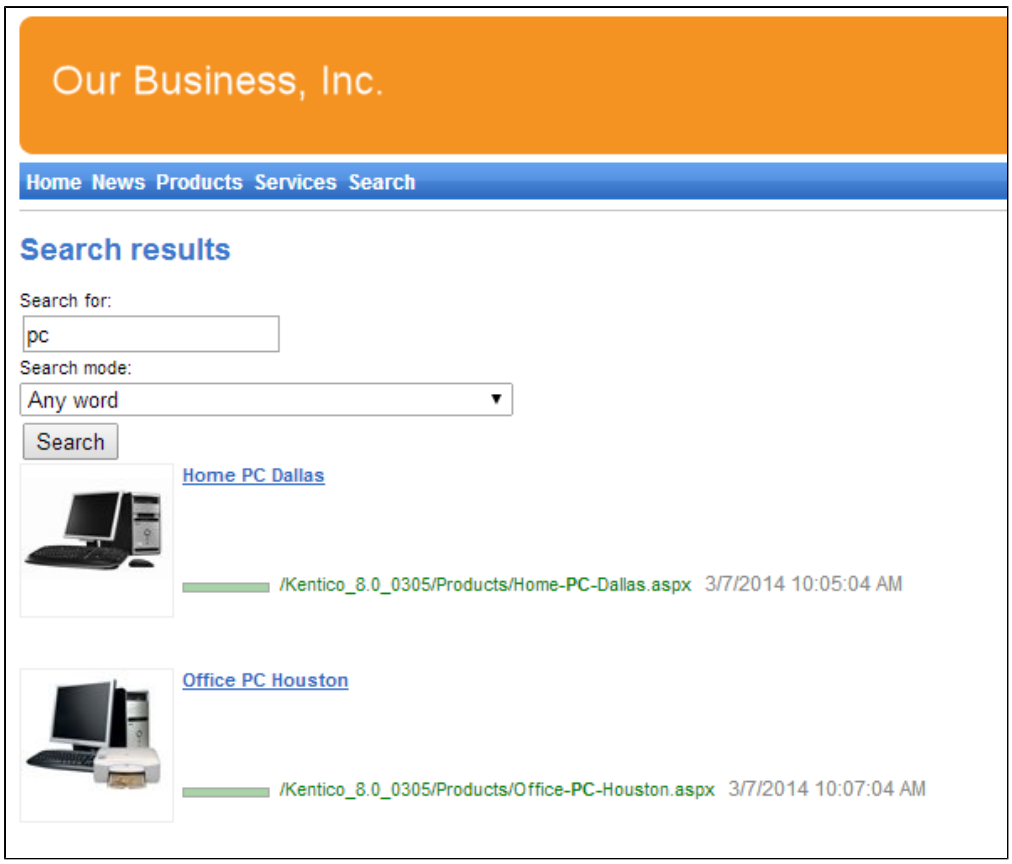
Once the system rebuilds the index, you can start using it on the website. The **Index info** section displays the current status of the index and other relevant information.



Adding the search page

1. Open the **Pages** application.
2. Select the root of the content tree (**My website**).
3. Click **New** (+).
4. Choose the **Page (menu item)** page type.
5. Type ***Search*** as the **Page name** and choose the **Use existing page template** option. Select the **General** category and the **Search** t emplate.
6. Click **Save** to create the page.
7. Click and drag the **Search** page to the end of the page list.
8. Switch to the **Design** tab and **Configure** (double-click) the **SearchDialogWithResults** web part.
9. Click **Select** next to the **Indexes** property and choose the **My website - Pages** search index created in the previous section.
10. Click **Save & Close**.

To try out the search functionality, view the /Search page in **Preview** 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 format of the search results

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

Adding a secured section for partners

Kentico provides a way to create secured site sections that can only be viewed by users who have a valid user name and password. This page describes how to create a logon web 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. Open the **Pages** application and select the root of the content tree (**My website**).
2. Click **New** (+)**.**
3. Choose the **Page (menu item)** page type.
4. Type ***Partners*** as the **Page name** and choose the **Use existing page template** option. Select the **Templates with editable regions** category and the **Simple text** page template.
5. Click **Save** to create the page.
6. On the **Page** tab, type the following text into the editable region: *This is a secured page for partners.*
7. Click **Save**.
8. Open the **Properties -> Security** tab of the **Partners** page.
9. Select ***Yes*** for the **Requires authentication** property in the **Access** section
10. Click **Save**.

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

Creating the logon page

Now build a page where users can sign in to the website and anonymous visitors can register as new users. Use a predefined page template, which you can further customize as needed.

1. In the **Pages** application, select the root of the content tree (**My website**).
2. Click **New** (+).
3. Choose the **Page (menu item)** page type.
4. Type ***Logon*** as the **Page name** and choose the **Use existing page template** option. Select the **Membership and security** category and the **Log-on page with registration form** page template.
5. Click **Save** to create the page.
6. Drag the Logon page to the end of the content tree.
7. Open the **Properties -> Navigation** tab.
8. Clear the **Show in navigation** and **Show in sitemap** check boxes.
9. Click **Save**.

Because of the page's navigation settings, the Logon page does not show up in the website's menu. You can use this configuration for pages that have a special purpose on the website, but are not part of the 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 the Kentico administration interface. However, you can configure each website to use its own custom logon page.

1. Open the **Settings** application.
2. Select the **Security & Membership** category in the settings tree.
3. Select **My website** in the **Site** drop-down menu.
4. Clear the **Inherit from global settings** check box next to the **Website logon page URL** setting and type in ***~/Logon.aspx***. This is the relative URL of the logon page that you added to the website.
5. Click **Save**.

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 the appropriate web parts to the website's master page.

1. Open the **Pages** application.
2. Select the root of the content tree (**My website**).

3. Switch to the **Design** tab.
4. Add the following web parts to the **zoneMain** web part zone and set their properties:

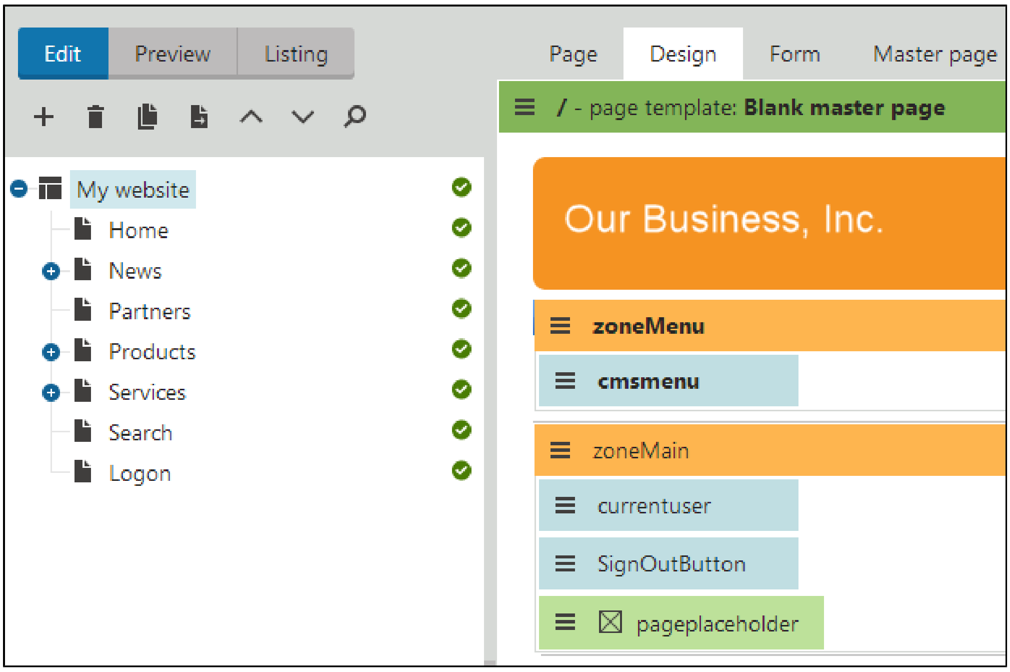
   Current user
   - **Content before**: <div style="float:right">

   Sign out button
   - **Content after**: </div>

   > The **<div>** tag encloses the Current user and Sign out button web parts and floats them to the right side of the page.

5. Set the following order for the web parts in **zoneMain** (you can drag web parts using your mouse):
   1. Current user
   2. Sign out button
   3. Page placeholder

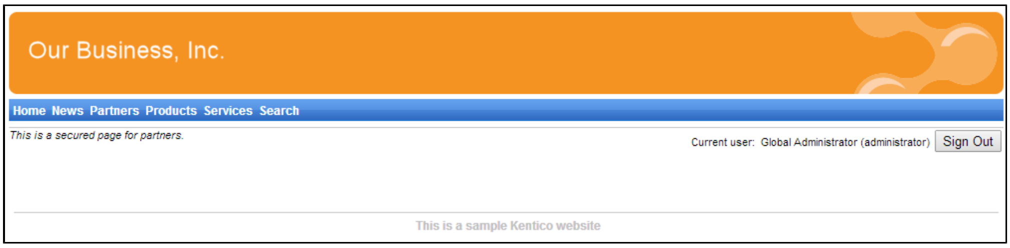
The **Sign out** button and the web part showing the current user's name are now visible for signed in users on all pages on the website.



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. Open the user menu on the right of the Kentico administration interface header, and select **Sign Out**.
2. Click **Partners** in the main menu. The page is restricted, so the website redirects you to the logon page.
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.



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

> Kentico 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.
>
> See Configuring permissions in the main documentation for more information.

Creating pages using ASPX templates

ASPX page templates in Kentico allow you to control every aspect of page processing. On an ASPX template, you can use standard ASP.NET controls, as well as controls and web parts provided by Kentico. You can also modify the code of the pages freely. However, in exchange for the customizability of pages, you lose the convenience of adding and configuring web parts and widgets directly from your browser.

Most Kentico Certified Developers prefer using the Portal engine over developing ASPX templates.

To develop ASPX templates, you need:

- Microsoft Visual Studio or an equivalent IDE
- knowledge of ASP.NET and C# or VB
- a supported web browser

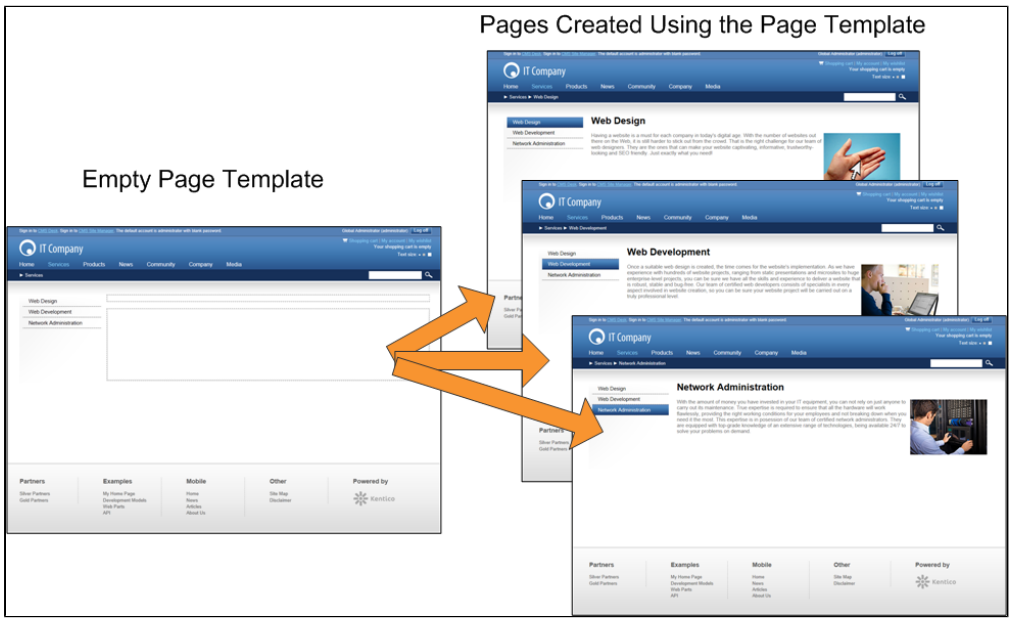
ASPX page templates

If you are familiar with ASP.NET development in Visual Studio, you can develop Kentico websites using ASPX page templates.

ASPX page templates are standard ASP.NET web forms. When you register ASPX templates in the system, users can create pages based on the templates and fill in content. The system internally uses the **aliasPath** URL parameter, which identifies the Kentico page whose content the template displays.

What is a page template?

Every web page is based on a page template. The page template can be specific for a single page (so called "ad hoc" page template) or re-used for any number of pages. The following picture shows examples of pages that use the same page template.

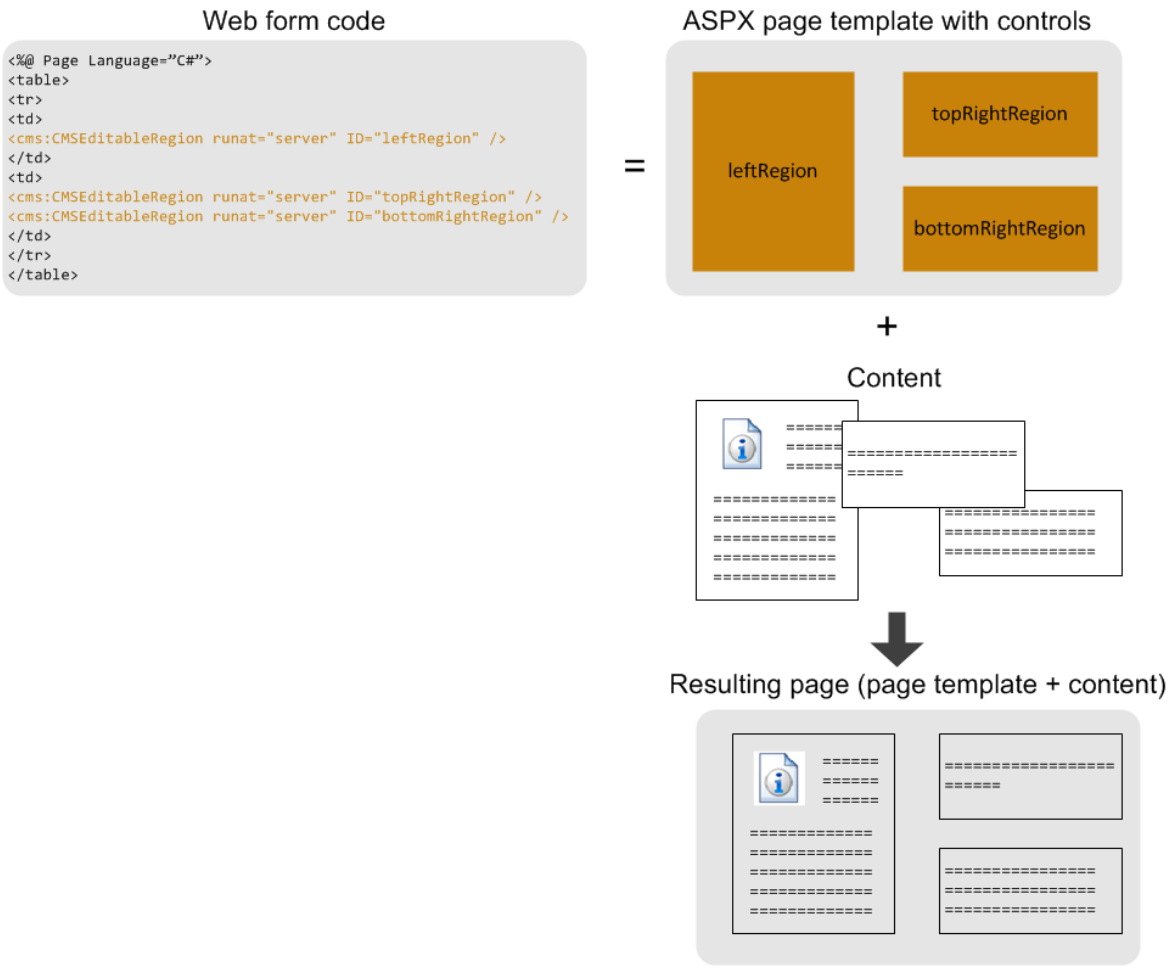


The pages have different content, but use the same header, menu content structure, and footer — they are based on the same page template. Templates allow you to quickly create multiple pages with the same design.

What do ASPX page templates consist of?

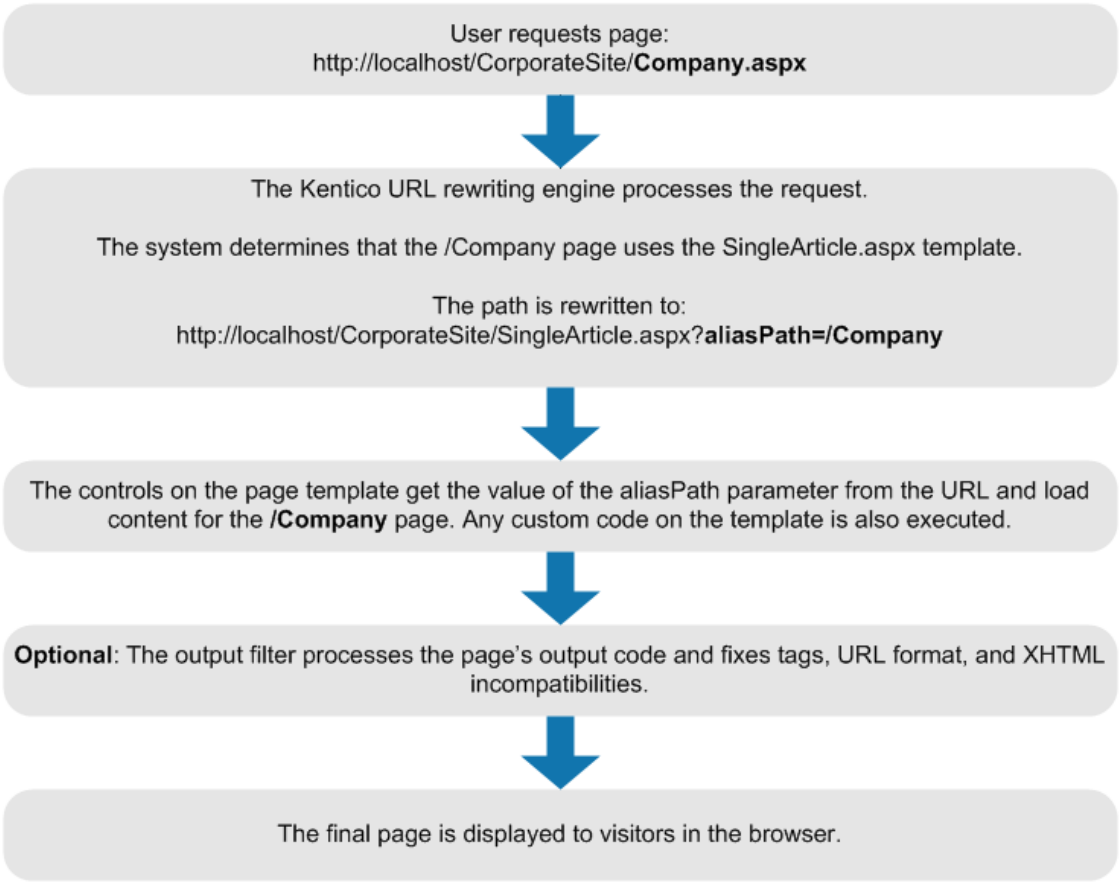
The content of page templates is a combination of static HTML code and ASP.NET server controls (or user controls) that render dynamic content. You can also use code behind (using either VB.NET or C#) to modify page behavior and add custom functionality.

The following figure illustrates how Kentico combines ASPX page templates with the content of individual pages to display the final result.



How does the system process ASPX page templates?

When a user requests a page, such as **~/Company.aspx**, the system internally calls the page template assigned to the given page with the **aliasPath** URL parameter. The parameter specifies what content (which page from the content tree) the page template displays to the user.



Kentico controls or web parts placed on the page template process the **aliasPath** parameter in the URL, and render the appropriate content automatically.

On the front-end, Kentico generates URLs in format **<domain>/Company.aspx,** which are more user-friendly and better for search engine optimization.

Creating a simple ASPX page template

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

Creating the web form

1. Open your web project in Visual Studio (using the **WebSite.sln** or **WebApp.sln** file).
2. Right-click the **CMSTemplates/CorporateSite** folder in the Solution Explorer and select **Add -> Add New Item**.

3. Create a new **Web form** named: *TwoColumnTemplate.aspx*
    - Check **Select master page**.
4. Click **Add**. The **Select a Master Page** dialog opens.
5. Choose the **CMSTemplates/CorporateSite** folder and select the default **Root.master** file.
6. Click **OK**.

Writing the ASPX code

1. Open the **Source** view of the new web form.
2. Add the following code inside the **<asp:Content>** element:

```
<table>
  <tr>
 <td style="width: 50%">
  <cms:CMSEditableRegion ID="txtLeft" runat="server" DialogHeight="400"
RegionType="HtmlEditor" RegionTitle="Left column" />
 </td>
 <td style="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 standard ASP.NET 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 in the Kentico administration interface on the **Page** tab of the **Pages** application. On the live site, the control renders the content entered into the editor.

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.

3. Edit the web form's code behind file (TwoColumnTemplate.aspx.cs).
4. Add a reference to the **CMS.UIControls** namespace:

```
using CMS.UIControls;
```

5. Modify the class declaration so that the web form inherits from **TemplatePage**:

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

6. Save the web form's files.

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

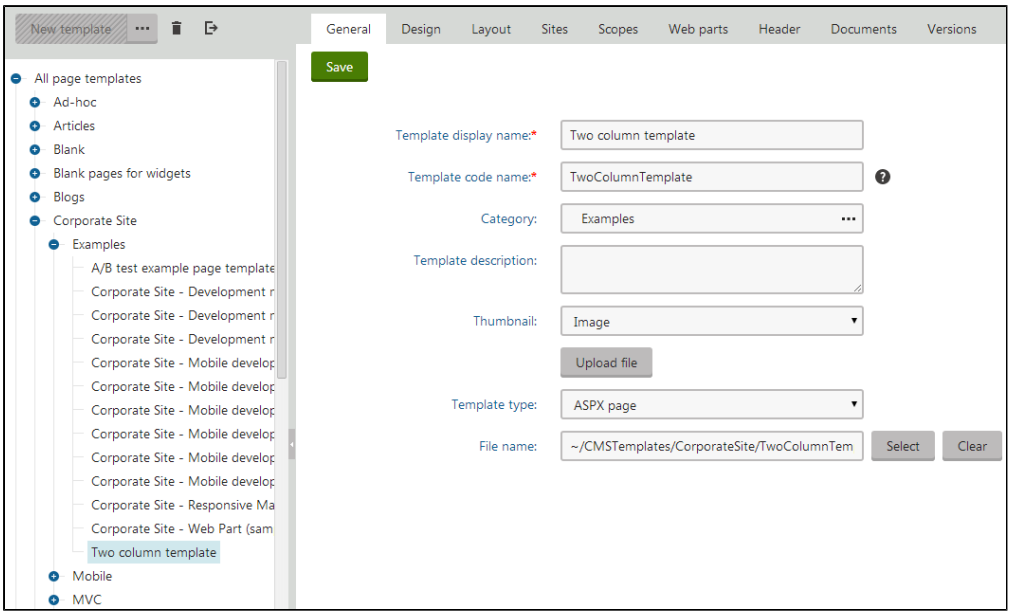
Keep in mind that the name of the class must be identical to the value of the **Inherits** attribute of the **<%@ Page %>** directive on the ASPX page. This is case sensitive.

Registering the web form as a page template

Now you need to register the web form as a page template in Kentico, so that it can be selected when creating pages.

1. Log in to the administration interface and open the **Page templates** application.
2. Select the **Corporate Site/Examples** category.
3. Click **New template**.
4. Type *Two column template* into the **Template display name** field.
5. Click **Save**.
6. Set the following values on the **General** tab:

- **Template type**: ASPX page
- **File name**: ~/CMSTemplates/CorporateSite/TwoColumnTemplate.aspx

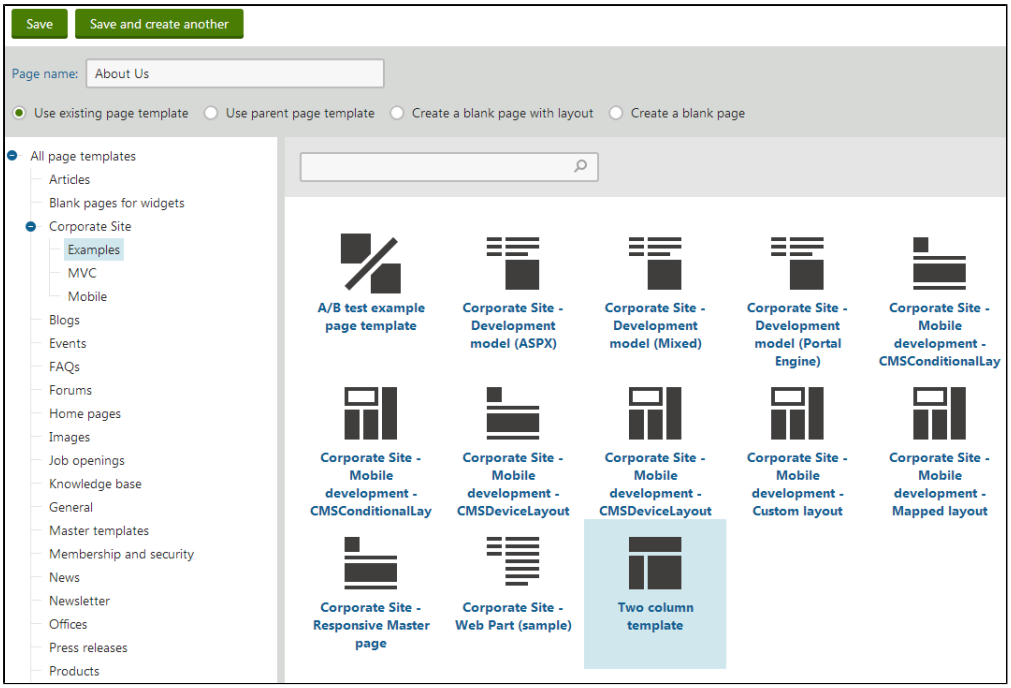


7. Click **Save**.
8. Switch to the **Sites** tab.
9. Click **Add sites**.
10. Choose the sites where you wish to use the page template (Corporate site) and click **Select**.

Creating a page based on the template

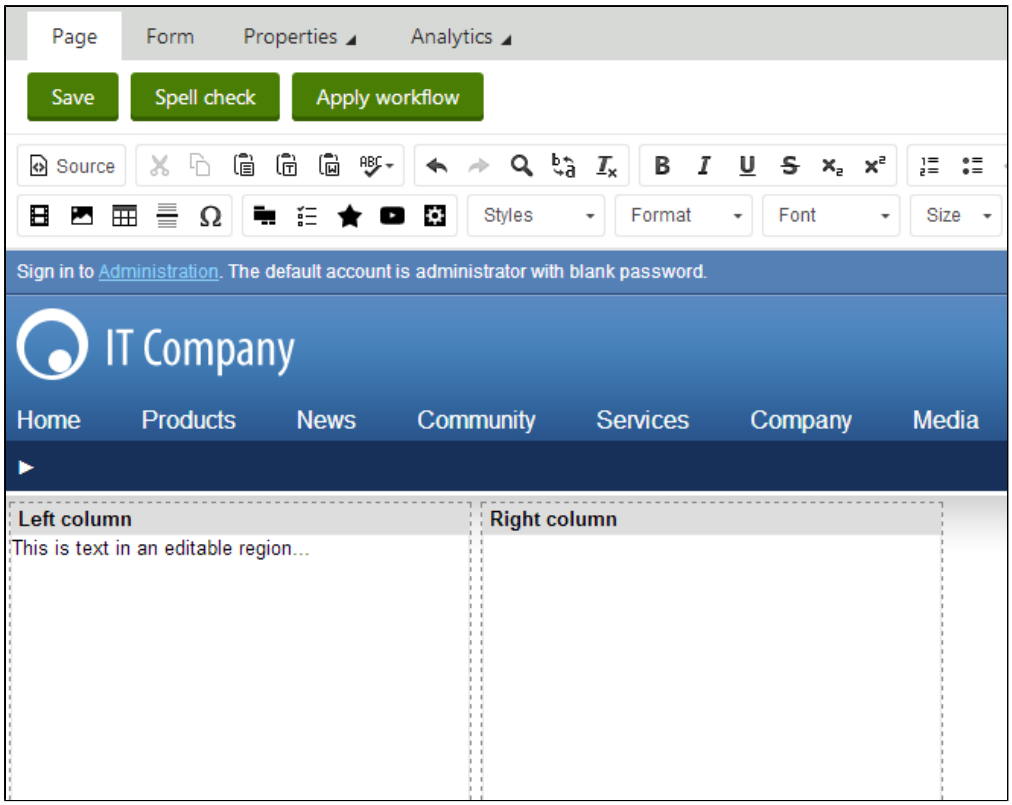
Content editors can now use the page template to create pages.

1. Open the **Pages** application.
2. Select **Corporate Site** (the root of the content tree).
3. Click **New** (+) above the tree.
4. Choose the **Page (menu item)** page type.
5. Type *About Us* as the **Page name** and choose the **Use existing page template** option.
6. Select the **Corporate Site/Examples** category and the **Two column template** page template.



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

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



You can now type in text into the regions and click **Save** to store the content of the page.

> **Moving pages**
>
> If you want to move the About Us page to a different position, you can drag the page to the desired location in the content tree, or click the **Move up** (^) and **Move down** (v) actions.

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.

- Define master pages 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:

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

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

> **Tip**: We recommend storing master pages in the **CMSTemplates** folder together with your ASPX page template files. This allows the system to export master pages along with your website when you deploy it to another instance of Kentico.

Creating master pages for ASPX templates

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

> **Important**
>
> - If you installed the Kentico project as a web application, you need to rename the **CodeFile** attribute on the first line to **Cod**

ebehind for the code example to be functional.
- The **CodeFile/Codebehind** attribute's value must 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="ltlTags" 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:

Control name	Description
ajaxToolkit:ToolkitScriptManager	Allows pages to use AJAX components. If required, the **CMSPortalManager** automatically loads the *ToolkitScriptManager*, but adding the control directly reduces overhead.
CMSPortalManager	Ensures the transferring of content between the database and editable regions. Also provides the management functionality needed for portal engine zones. The CMSPortalManager must be placed inside a standard PlaceHolder control.

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 need to modify the code behind file of your master pages according to the following steps:

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

```
using CMS.UIControls;
```

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

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

> Master pages of ASPX templates must always inherit from the **TemplateMasterPage** class.

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

> Adjust the value of the **PageManager** property according to the ID of the *CMSPortalManager* control placed on the master page.

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

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

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

Adding portal engine functionality to ASPX templates

When developing or maintaining a website using ASPX page templates, one of the drawbacks is that you need to manually modify the code of pages whenever you wish to change the design. You can add flexibility to ASPX templates by defining areas that are editable directly through the browser in the **Pages** application, just like when using the Portal engine development model. To learn more about portal engine features, please read the version of this tutorial dedicated to the portal engine.

The following example demonstrates how to create an ASPX page template with zones that users can design via the portal engine:

Writing the ASPX code

1. Open your web project in Visual Studio (using the **WebSite.sln** or **WebApp.sln** file).
2. Rightclick the **CMSTemplates\CorporateSite** folder in the Solution Explorer and select **Add -> Add New Item**.
3. Create a new **Web form** named: *TwoZones.aspx*
    - Check the **Select master page** box.
4. Click **Add**. The **Select a Master Page** dialog opens.
5. Choose the **Root.master** page from the **CMSTemplates/CorporateSite** folder and click **OK**.
6. Open the **Source** view of the new ASPX page and place the following inside the **<asp:Content>** element:

```
<cms:CMSPagePlaceholder ID="plcZones" runat="server">
    <LayoutTemplate>
        <table style="width:100%">
            <tr>
              <td style="width: 50%">
                <cms:CMSWebPartZone ID="zoneLeft" runat="server" />
              </td>
              <td style="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 CSSbased layout applied through HTML elements (for example ,) 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 pages based on the page template on the **Design** tab of the **Pages** application.When web part or widget content is added to 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 **CMS PortalManager** control, which is located on the Root.master page.

7. Switch to the code behind file (**TwoZones.aspx.cs**) and add a reference to the **CMS.UIControls** namespace:

```
using CMS.UIControls;
```

8. Modify the class definition to inherit from the **TemplatePage** class:

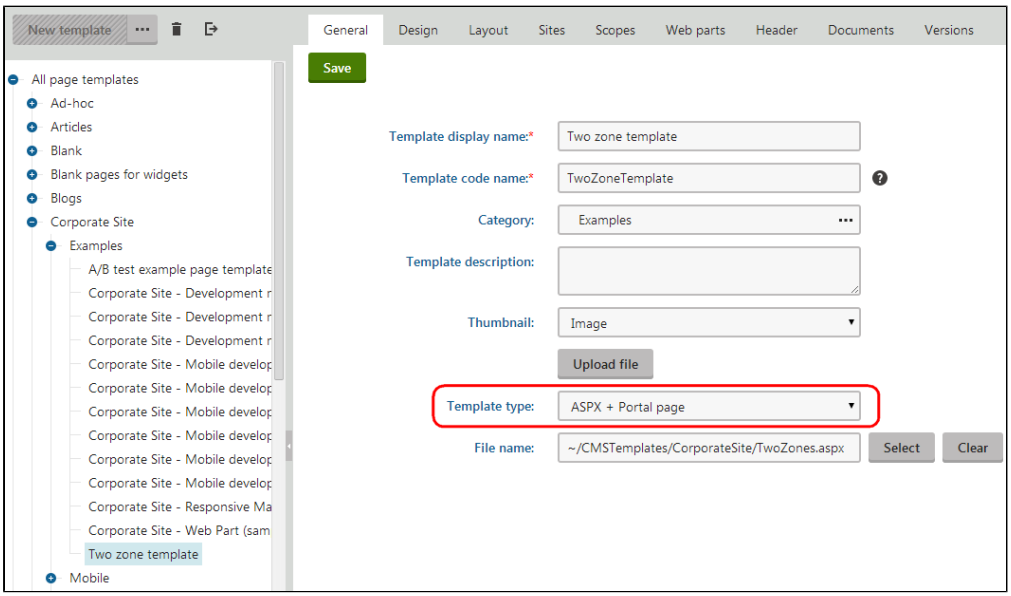
```
public partial class CMSTemplates_CorporateSite_TwoZones : TemplatePage
```

9. Save the web form's files.

You can now use the web form as a page template in Kentico.

Registering the ASPX page as a page template

1. Log in to the Kentico administration interface and open the **Page templates** application.
2. Select the **Corporate Site/Examples** folder.
3. Click **New template** and type ***Two zone template*** into the **Template display name** field.
4. Click **Save**. The system creates the template and displays its **General** tab.
5. Select the ***ASPX + Portal page*** option for the **Template type** property. This is necessary in order for the **Design** tab to be available when editing pages using the template in the **Pages** application.



6. Enter the following path into the **File name** field: ***~/CMSTemplates/CorporateSite/TwoZones.aspx***
7. **Save** the changes.
8. 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.

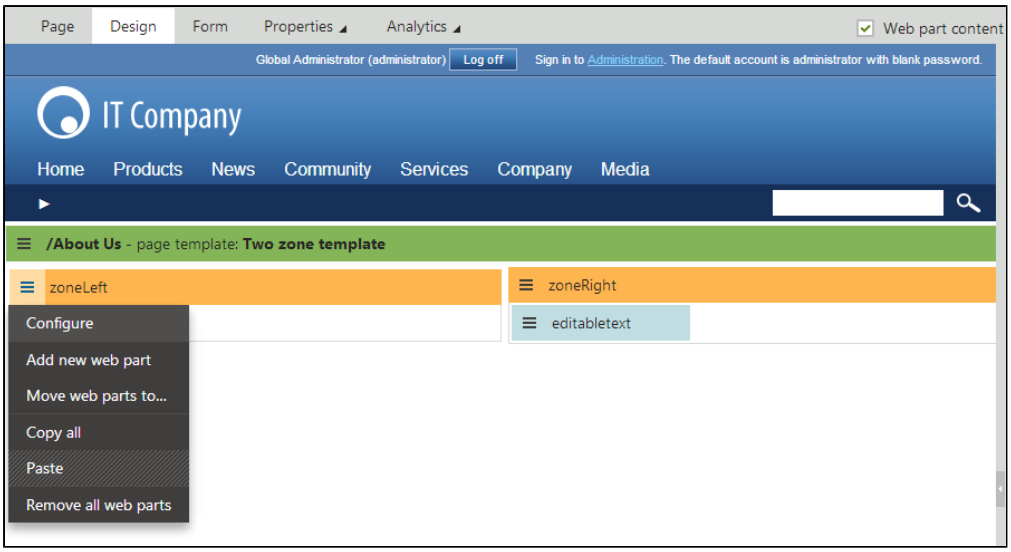
1. Open the **Pages** application.
2. Select the **About Us** page and switch to the **Properties -> Template** tab.
3. Click **Select** and choose the ***Corporate Site/Examples/Two zone template*** page template from the catalog.
4. Click **Save** to confirm the page template change.
5. Refresh your 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 web parts.
6. Drag the **Editable text** web part from the toolbar into **zoneRight**
7. Double-click the web part header in the zone and set the following properties:
    - **Design -> Editable region title**: Right text
    - **Design -> Editable region height**: 400

8. Click **Save & Close**.

This web part provides a text area on the page that users can edit on the **Page** tab of the **Pages** application, just like the editable regions in the previous example. The template allows you to build the design of the page using a browserbased interface. Each web part zone may contain any number of web parts.

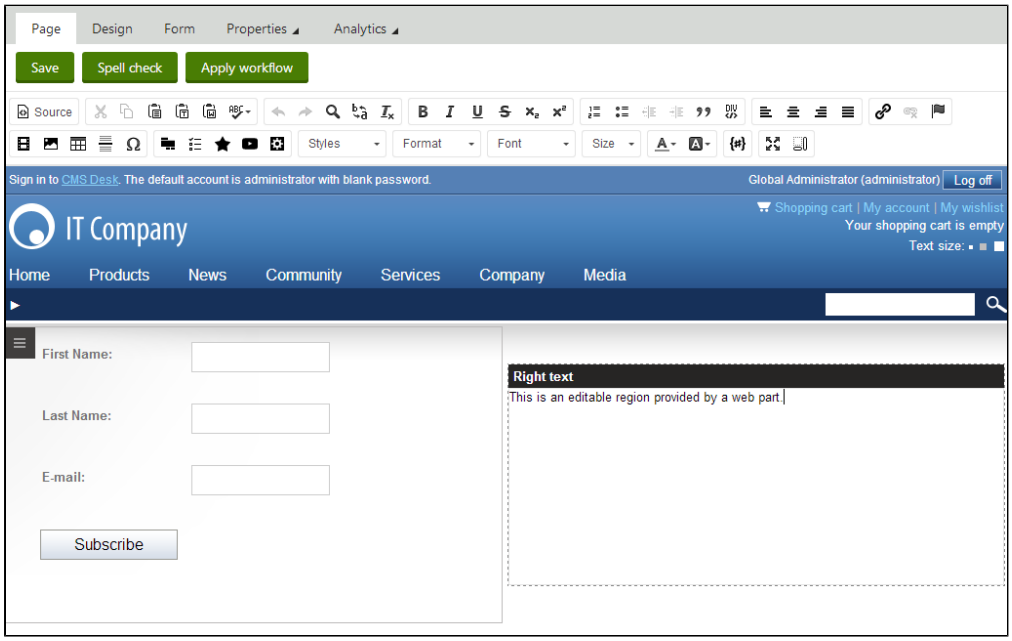
You may also configure zones to use various types of widgets, which are objects similar to web parts, but allow page customization by different kinds of website users, not just the administrators or designers.

1. Expand the menu ( ) of **zoneLeft** and select **Configure**.



2. Switch the **Widget zone type** property from ***None*** to ***Customization by page editor***.
3. Click **Save & Close**. The zone now serves as a widget zone for page editors.
4. Switch to the **Page** tab
5. Type some content into the editable text region displayed by the web part on the right and click **Save**.
6. Open the menu of the editor widget zone (click ☰) and click **Add new widget**. 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
7. Click **Save & Close** to add the widget and then **Save** the page.

The widget provides a form which users can 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 userfriendliness of the portal engine.

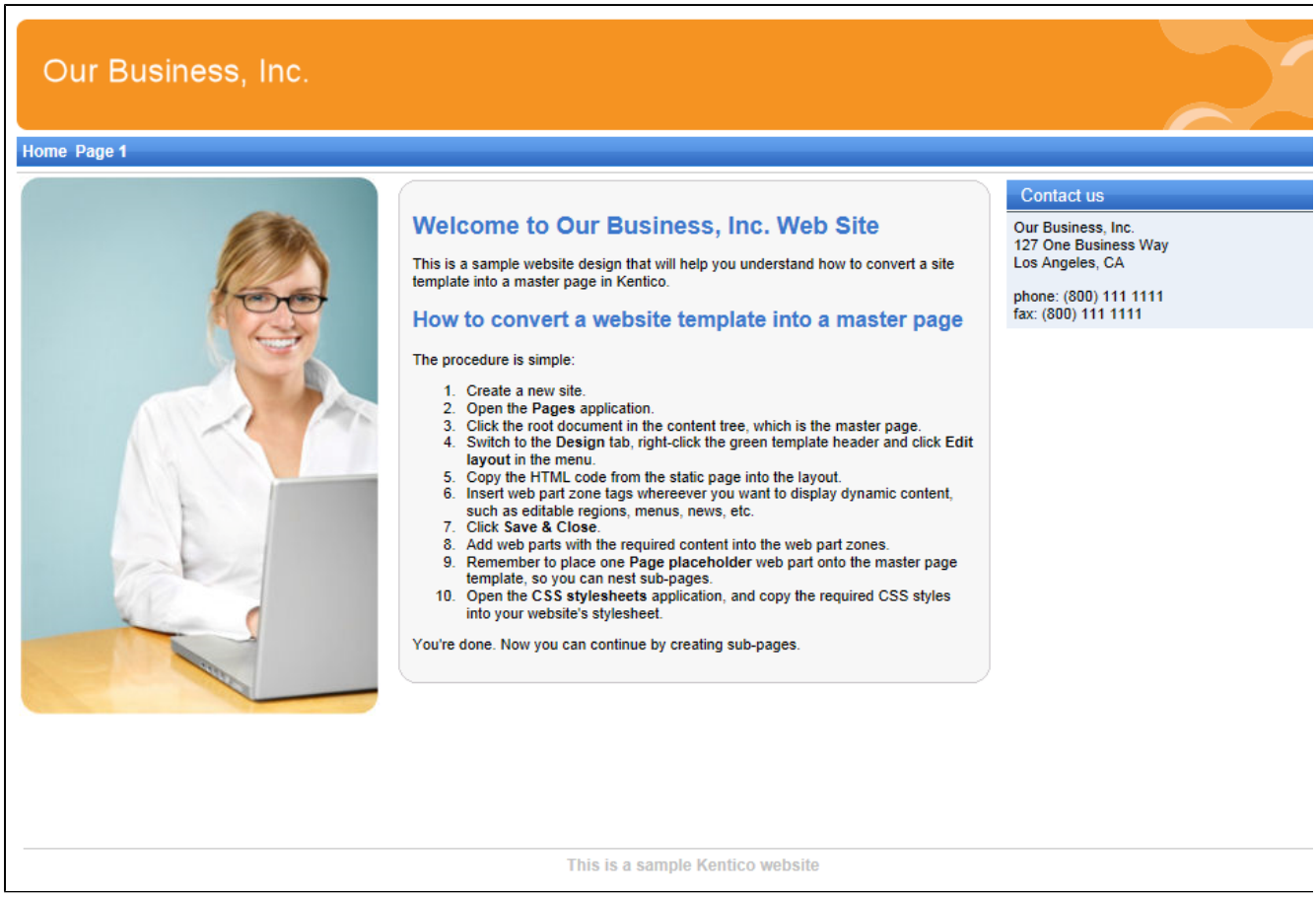
Walkthrough - Creating a new site using ASPX templates

This part of the tutorial guides you through the creation of a simple website. You will learn how to:

- Define site structure and design
- Create your own page templates and pages

The tutorial uses a static website template that is similar to what a developer gets from a graphic designer.

> **Click to download the sample web template**
>
> The template consists of the *home.htm* file, a styles folder and an *app_themes* folder with images.



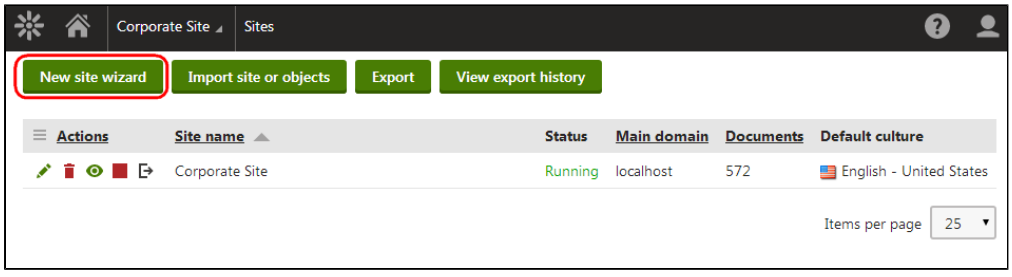
Creating a new website using the New site wizard (ASPX)

The following pages assume that you have previously installed the sample Corporate Site (running on the *localhost* domain). Leave the existing website and add a new site running under the *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 **Corporate Site** site in the **Sites** application. The built-in web server doesn't support any domain other than localhost, so you need to use the ***localhost* domain** ag ain for the new site.

1. Log in to the Kentico administration interface as ***administrator*** and open the **Sites** application.

2. Click **New site wizard**.



The New site wizard opens.

3. Select **Create a new site using a wizard** and click **Next**.

4. Enter the following details for the website:

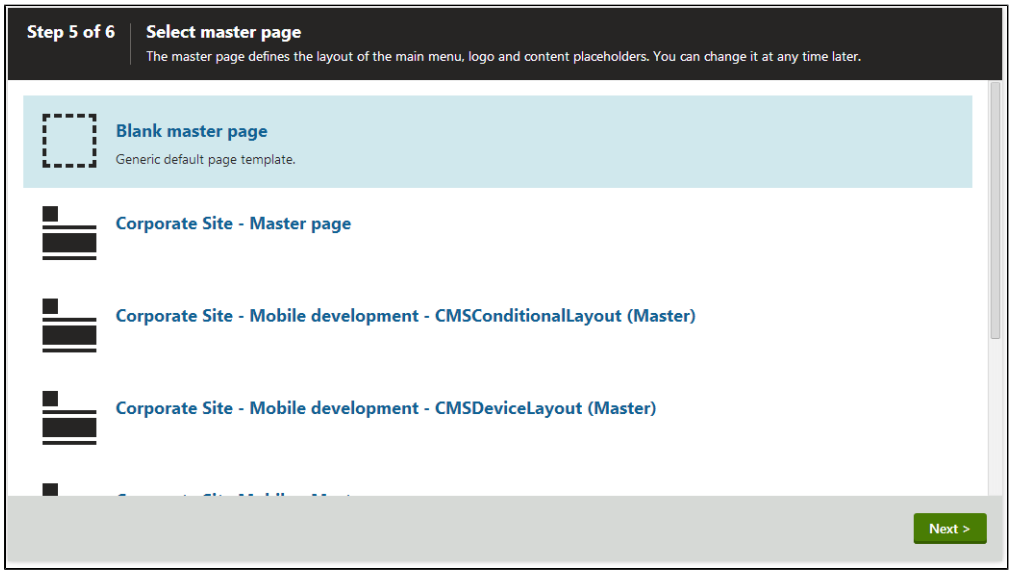
- **Site display name**: My website
- **Site code name**: mysite
- **Domain name**: 127.0.0.1 (if you are using Visual Studio built-in web server, set the Domain name value to *localhost)*
- **Site culture**: English - United States (the default culture determines how the website displays date, time and numeric values based on the different culture-specific format)

Click **Next**.

5. The third step of the wizard allows you to select which objects the system imports into the new site. **Do not change anything** and click **Ne xt**.

The fourth step displays the progress of the object import.

6. Choose a master page template for the website. You can change it later at any time. For now, select the **Blank master page**.



Click **Next**. This concludes the initial process of creating the website.

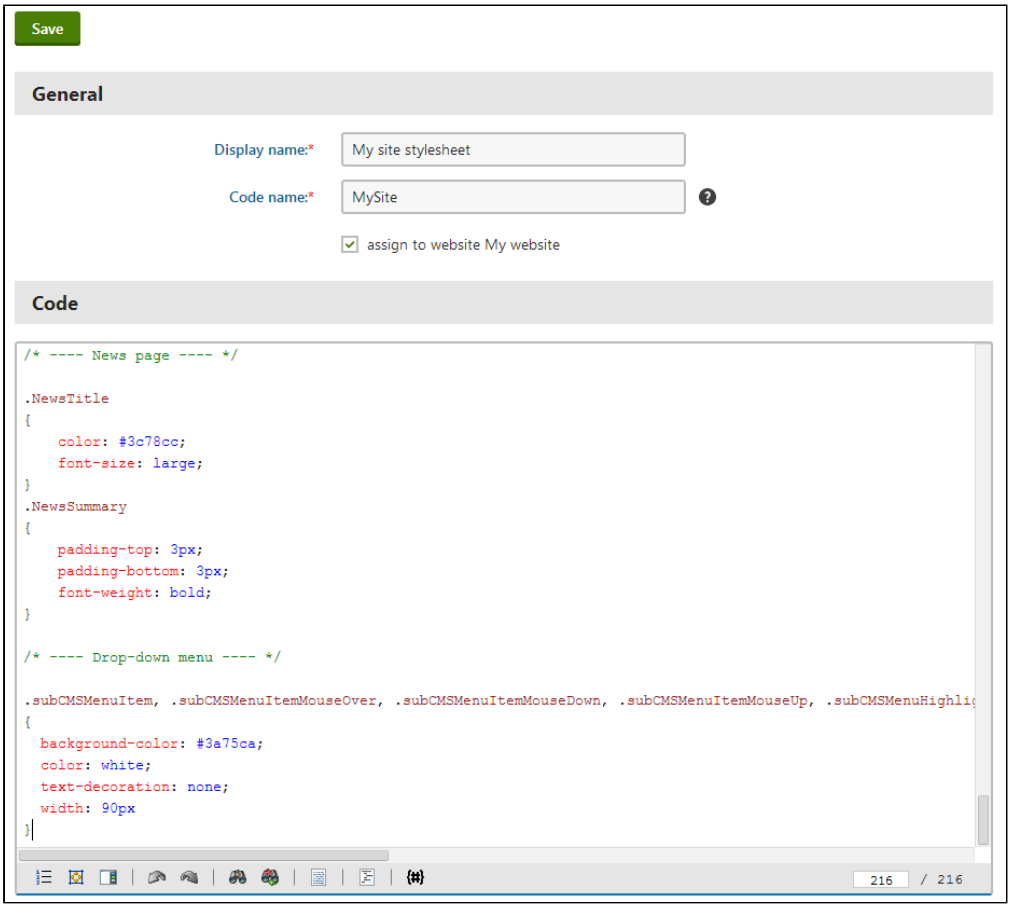
7. Click **Edit your new website**.

The system opens a new tab with the Kentico administration interface using the 127.0.0.1 domain. You need to sign in again (user name **ad ministrator,** blank password) since authentication is not shared over different domains by default.

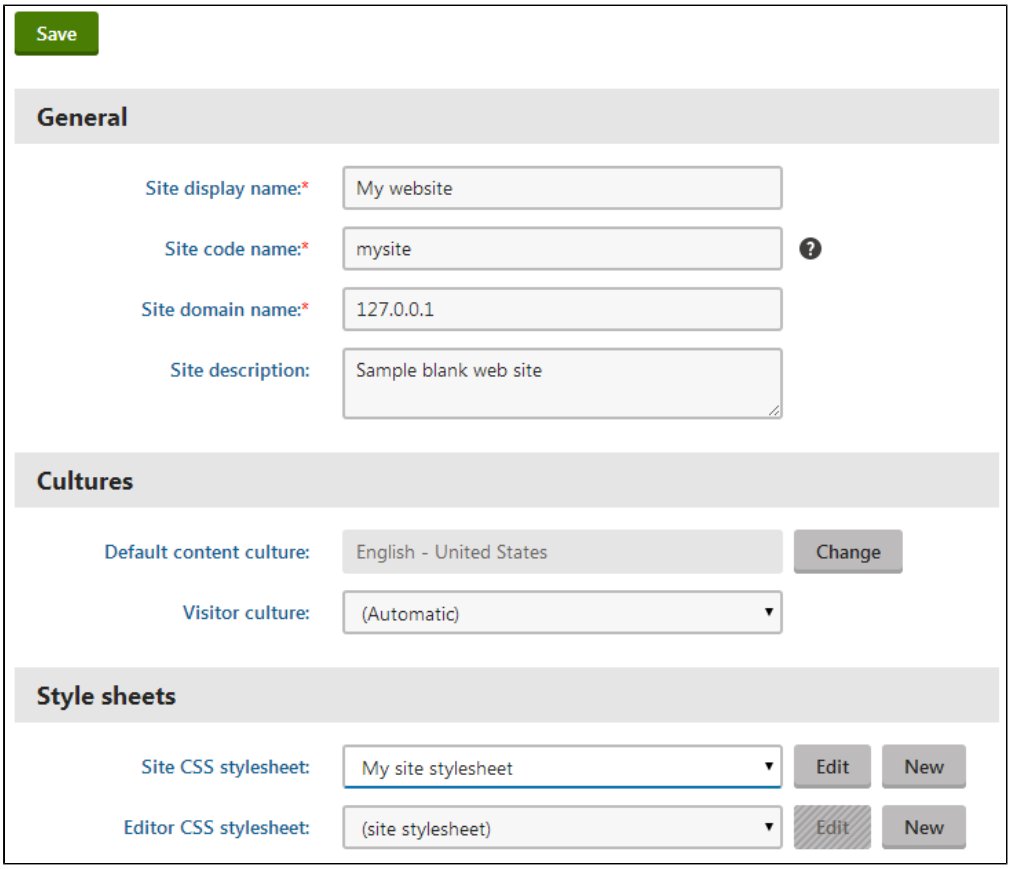
Creating a CSS stylesheet (ASPX)

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

1. Open the **CSS stylesheets** application.
2. Click **New CSS stylesheet**.
3. Enter the following values:
    - **Display name**: My site stylesheet
    - **Code name**: MySite
    - **Code**: copy and paste all CSS code from the Sample web template - **SampleWebTemplate\Styles\main.css**



4. Click **Save**.
5. Switch to the **Sites** tab and assign the stylesheet to **My website**.
6. Open the **Sites** application and edit ( ) **My website**.
7. On the **General** tab, select ***My site stylesheet*** as the **Site CSS stylesheet**.



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 **CMS\App_Themes** folder in your web project (*c:\inetpub\wwwroot\Kentico* by default).

The folder contains graphics for the website template. The App_Themes location ensures that the images are exported as part of the website if you decide to move the website in the future. 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**
>
> The image paths in the sample CSS stylesheet already 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 that you need to link to files in the App_Themes folder like in the example below:
>
> */app_themes/mysite/images/imagename.gif*

Opening and configuring the web project (ASPX)

Open your Kentico web project in Visual Studio. Use the **WebSite.sln** or **WebApp.sln** file in the website installation directory (*c:\inetpub\wwwroot\Kentico* by default).

Adding Kentico controls to the Visual Studio Toolbox

To make it easier to work with Kentico 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 (for example ***Kentico***) and press **Enter**.
4. Right-click the new tab and select **Choose Items**.
5. Click **Browse** on the **.NET Framework Components** tab of the **Choose Toolbox Items** dialog.
6. Navigate to the **CMS\bin** folder under your website.
7. Select the ***CMS.Controls.dll*** library.
8. Click **Open** and then **OK**.

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

Developing the master page (ASPX)

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.

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

> 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.

Adding the master page

1. Right-click the **MySite** folder, select **Add -> Add New item**.
2. Create a **Master page** named ***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
4. 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="ltlTags" 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" />
 </asp:PlaceHolder>
 </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.

5. Open the **home.htm** file (from the Sample web template) and copy the HTML code from inside the **<body>...</body>** tags. Paste this code into the body of the master page after the **<asp:PlaceHolder>** control.
6. Delete all code in the **<!-- main content --> ... <!-- /main content -->** section and replace it with the following control:

```
<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">
 <asp:PlaceHolder runat="server" ID="plcManagers">
  <ajaxToolkit:ToolkitScriptManager ID="manScript" runat="server"
EnableViewState="false" ScriptMode="Release" />
     <cms:CMSPortalManager ID="CMSPortalManager1" runat="server" />
 </asp:PlaceHolder>

 <div class="MainDiv">
       <!-- logo -->
       <br />
       <div class="Logo">
        
       </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 Kentico website
       </div>
    </div>
    </form>
</body>
```

Adjusting the master page code behind

1. Edit the code behind of the master page (**MyMaster.master.cs**).
2. Add a reference to the **CMS.UIControls** namespace:

```
using CMS.UIControls;
```

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

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

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

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

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

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

6. Save the master page files.

Continue editing the master page according to the instructions in Creating the main menu (ASPX).

Creating the main menu (ASPX)

Now 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.

> If you prefer a drop-down menu based on CSS styles and UL/LI elements, you can try using the CMSListMenu later. See the 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 ** ** 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
Path	/%	Configures the menu to display pages starting from the root of the site structure.
Layout	Horizontal	Sets a horizontal layout for the menu.
CSSPrefix	;sub	Allows you to add prefixes before the names of the CSS classes applied to the menu. The ;sub value uses unmodified class names for the main (first) menu level and the sub prefix for the second level and all other sub-levels.
Cursor	Pointer	Specifies the type of mouse cursor displayed when a user hovers over the menu.

4. Save the changes.

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>
```

The master page is now prepared and you can assign it to the site's ASPX templates. Continue with Developing the Home page (ASPX)

Kentico Controls and Web Parts

Kentico 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.

Developing the Home page (ASPX)

This page describes how to create the home page of the website. The process consists of the following steps:

Preparing the ASPX source file

1. Edit your web project in Visual Studio
2. Right-click the **CMSTemplates/MySite** folder in the Solution Explorer and click **Add -> Add New Item**.
3. Create a **Web Form** named ***Home.aspx*** and check **Select master page**.
4. Click **Add** and choose the ***MyMaster.master*** page from the **CMSTemplates/MySite** folder.
5. Open the **home.htm** file (from the Sample web template) and copy the HTML code from inside the **<!-- main content -->** section. Paste this code inside the **<asp:Content>** element of the **Home.aspx** file.
6. 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 style="vertical-align:top;">
                <!-- left column -->
                <td style="width:280px" class="HomePageLeftColumn">
                </td>
                <!-- center column -->
                <td style="padding: 3px 5px 0px 5px;width:450px;">
                    <!-- center box -->
                    <table cellspacing="0" cellpadding="0" border="0"
class="ContainerWithCorners" width="100%">
                        <tr class="ContainerWithCornersRow">
                            <td class="ContainerWithCornersTopLeft"> </td>
                            <td class="ContainerWithCornersTop"> </td>
                            <td class="ContainerWithCornersTopRight"> </td>
                        </tr>
                        <tr>
                            <td class="ContainerWithCornersLeft"> </td>
                            <td class="ContainerWithCornersContent" valign="top">

                            </td>
                            <td class="ContainerWithCornersRight"> </td>
                        </tr>
                        <tr class="ContainerWithCornersRow">
                            <td class="ContainerWithCornersBottomLeft"> </td>
                            <td class="ContainerWithCornersBottom"></td>
                            <td class="ContainerWithCornersBottomRight"> </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>
<!-- /main content -->

</asp:Content>
```

Replace the text with editable regions so that content editors can manage the page in the Kentico administration interface.

1. 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"
style="width: 100%;">
    <tr class="ContainerWithCornersRow">
        <td class="ContainerWithCornersTopLeft"> </td>
        <td class="ContainerWithCornersTop"> </td>
        <td class="ContainerWithCornersTopRight"> </td>
    </tr>
    <tr>
        <td class="ContainerWithCornersLeft"> </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"> </td>
    </tr>
    <tr class="ContainerWithCornersRow">
        <td class="ContainerWithCornersBottomLeft"> </td>
        <td class="ContainerWithCornersBottom"></td>
        <td class="ContainerWithCornersBottomRight"> </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, you will configure this area to allow content editors to customize the design of the Home page.

2. 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.
3. Set the following properties of the **CMSEditableRegion** control:
    - **ID**: txtRight
    - **DialogHeight**: 280
    - **RegionTitle**: Right content
    - **RegionType**: HtmlEditor
4. Switch to the code behind of the home page (**Home.aspx.cs**) and add a reference to the **CMS.UIControls** namespace:

```
using CMS.UIControls;
```

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

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

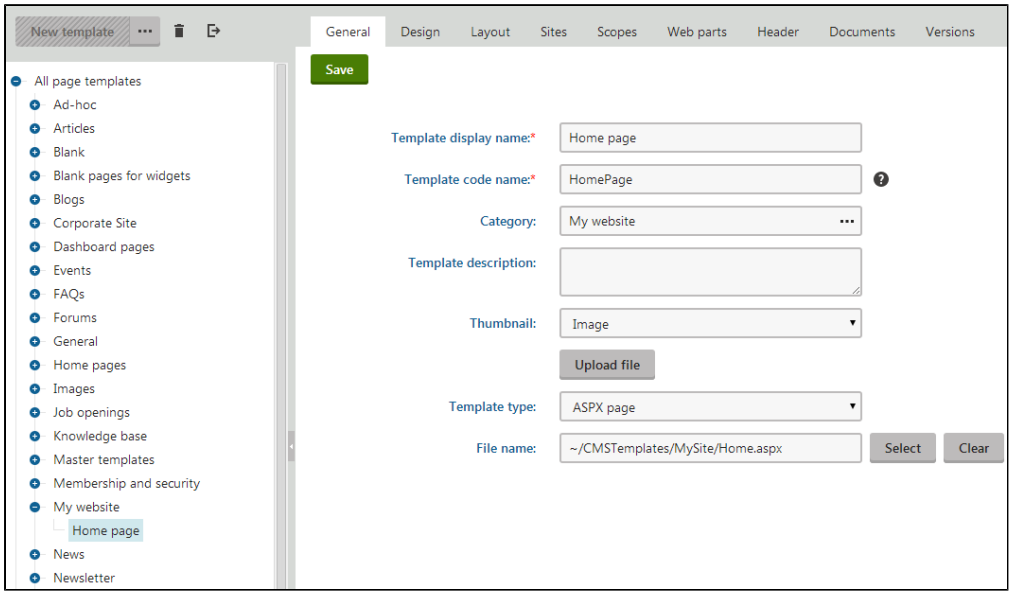
6. Save the home page files.

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.

1. Open Kentico in a web browser and log in to the administration interface (default username **administrator** with an empty password).
2. Open the **Page templates** application.
3. Click **...** next to the **New template** button and select **New category**.
4. Type ***My website*** into the **Category display name** field.
5. Click **Save.**
6. Click **New template** and type ***Home page*** into the **Template display name** field.

7. Click **Save**.
8. Set the following values on the **General** tab:
    - **Template type**: ASPX + Portal page
    - **File name**: ~/CMSTemplates/MySite/Home.aspx
9. Click **Save**.
10. Switch to the **Sites** tab and assign the page template to **My website**.

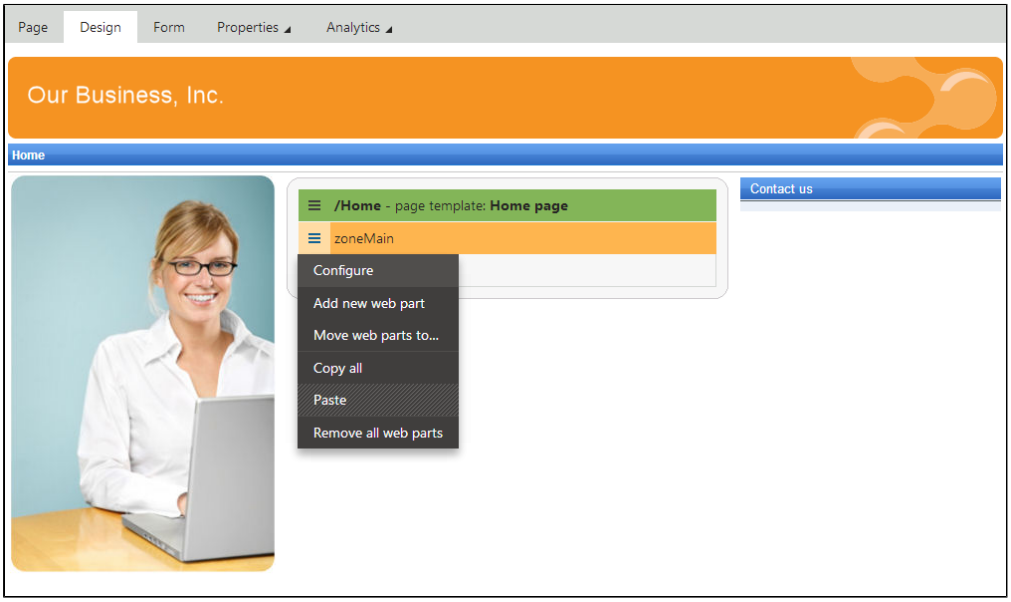


Adding the Home page

1. Open the **Pages** application.
2. Select the root of the content tree (**My website**).
3. Click **New** (+).
4. Choose the **Page (menu item)** page type.
5. 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.
6. Click **Save** to create the page.

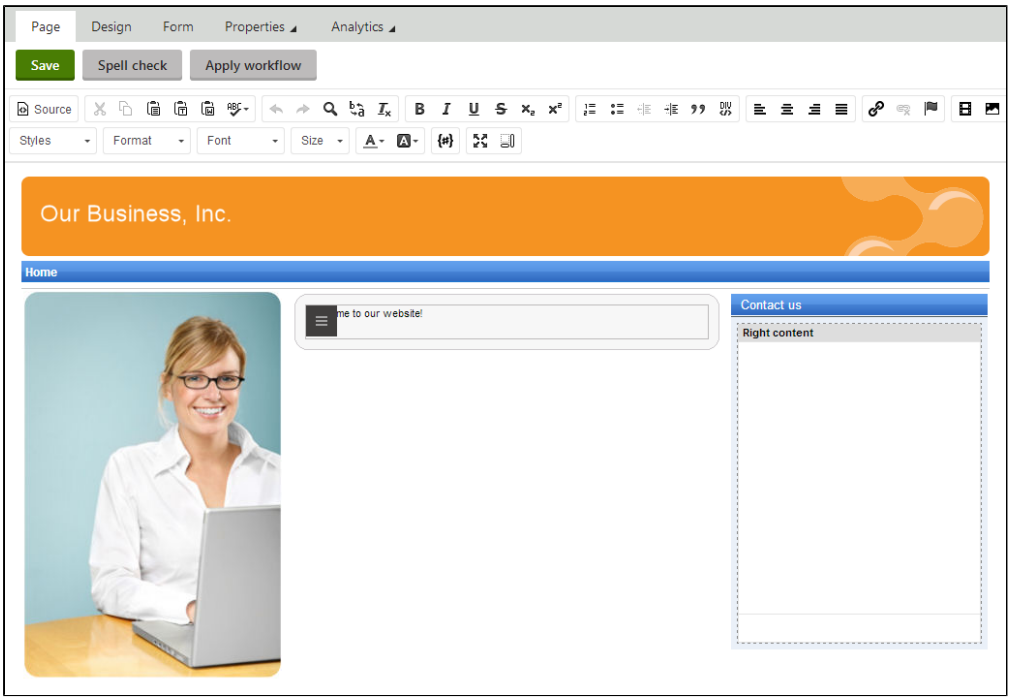
Editing the content of the Home page

1. With the Home page selected, switch to the **Design** tab. Here you can see the editable area that you defined in the code of the page template.
2. Right-click the header of the **zoneMain** zone and select **Configure** in the menu.



3. Switch the value of the **Widget zone type** property from ***None*** to ***Customization by page editor*** and click **Save & Close**.

4. Open the **Page** tab, expand the widget zone's menu ( ) and click **Add new widget**.
    a. Select the **General -> Text** widget.
    b. Click **Select**.
    c. Type ***Welcome to our website!*** into the **Text** property
    d. Click **Save & Close**.



> 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.

5. Type the following text into the **Right content** editable region: ***Call 800 111 2222***
6. Click **Save**. You always need to click *Save* to confirm changes made to the text of editable regions or widget content on the *Page* ta b.

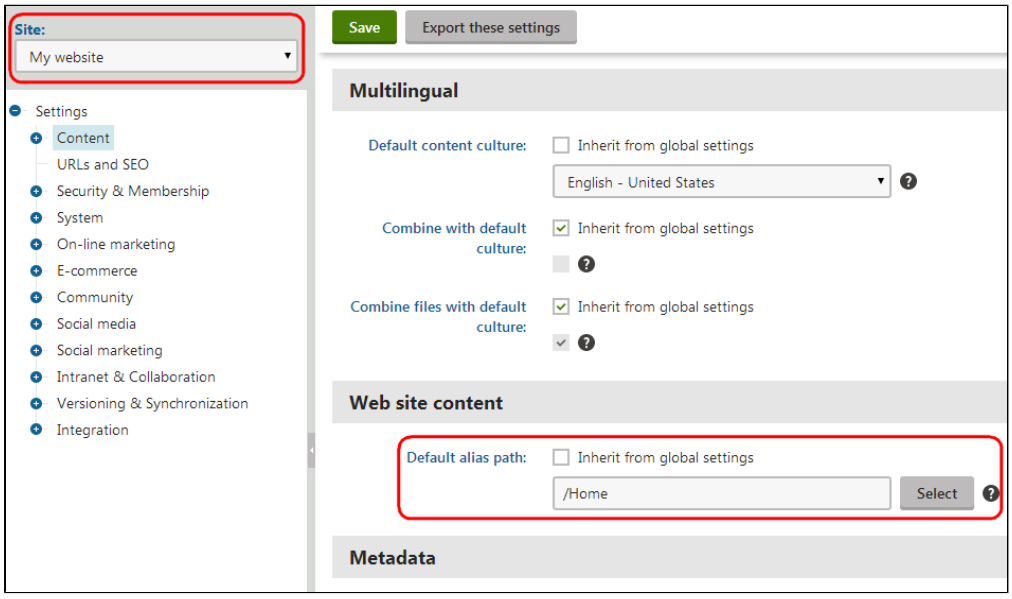
Switch to **Preview** mode to see how the home page of your website appears to visitors.

Choosing the website's home page

When a visitor arrives at 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. Open the **Settings** application.
2. Select the **Content** category in the settings tree.
3. Select **My website** in the **Site** drop-down.
4. Clear the **Inherit from global settings** check box next to the **Default alias path** setting.
5. Type in ***/Home***, which is the alias path of your new home page.
6. Click **Save**.



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

Developing the News page (ASPX)

Create the News section of the website.

Preparing the ASPX source file

1. Edit your web project in Visual Studio.
2. Right-click the **CMSTemplates/MySite** folder in the Solution Explorer and click **Add -> Add New Item**.
3. Create a **Web Form** named ***NewsPage.aspx*** and check **Select master page**.
4. Click **Add** and choose the ***MyMaster.master*** page from the **CMSTemplates/MySite** folder.
5. Drag the following controls inside the **<asp:Content>** element of the news page:
    - CMSBreadCrumbs
    - CMSRepeater
6. 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 pages of the cms.news type.
TransformationName	cms.news.preview	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.

```
<cms:CMSRepeater ID="CMSRepeater1" runat="server" ClassNames="cms.news"
TransformationName="cms.news.preview"
SelectedItemTransformationName="cms.news.default" ItemSeparator="<hr />" />
```

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

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

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

```
using CMS.UIControls;
```

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

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

10. Save the news page files.

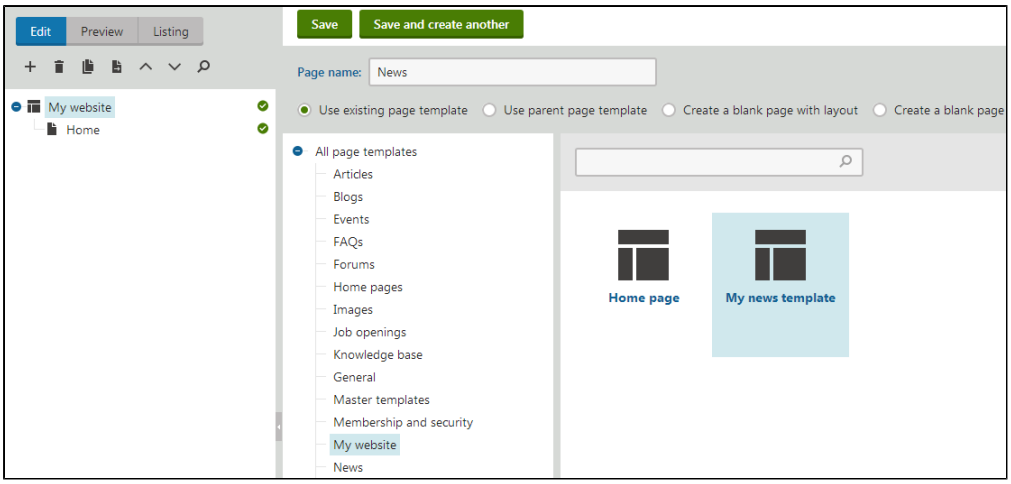
Registering the page template

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

1. Switch to the Kentico administration interface in your browser.
2. Open the **Page templates** application.
3. Select the **My website** category.
4. Click **New template** and type ***My news template*** into the **Template display name** field.
5. Click **Save**.
6. Set the following values on the **General** tab:
   - **Template type**: ASPX page
   - **File name**: ~/CMSTemplates/MySite/NewsPage.aspx
7. Click **Save**.
8. Switch to the **Sites** tab and assign the page template to **My website**.

Adding the news section

1. Open the **Pages** application.
2. Select the root of the content tree (**My website**).
3. Click **New** (+).
4. Choose the **Page (menu item)** page type.
5. 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.

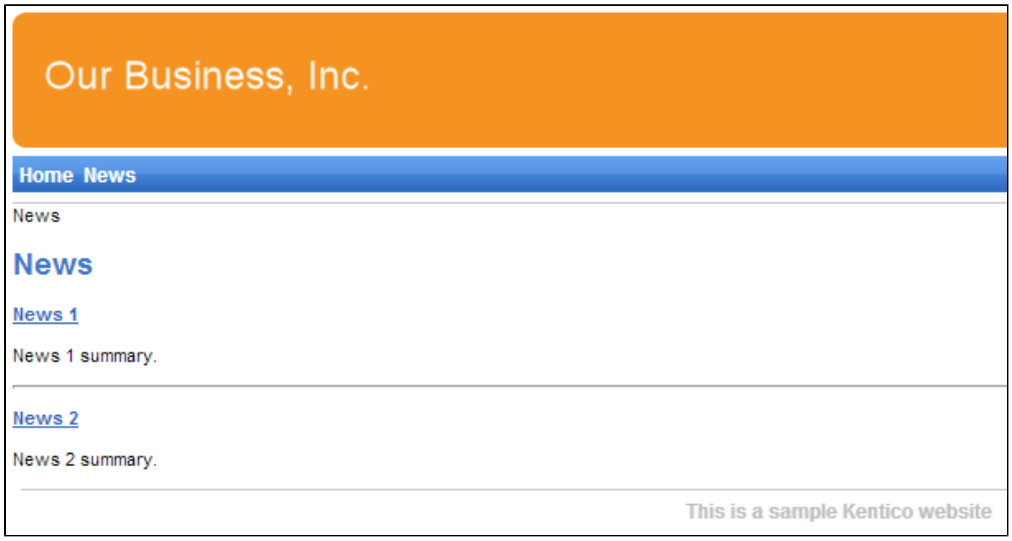


6. Click **Save** to create the page.
7. Select the **News** page in the content tree
8. Click **New** (+) and choose the **News** page type.
9. Fill in the news page fields with the following values:
    - **News Title**: News 1
    - **Release Date**: click **Today**
    - **News Summary**: News 1 summary.
    - **News Text**: News 1 text.
    - **Publish from/Publish to**: leave the fields blank

10. Click **Save and create another** and enter the following values:
    - **News Title**: News 2
    - **Release Date**: click **Today**
    - **News Summary**: News 2 summary.
    - **News Text**: News 2 text.
    - **Publish from/Publish to**: leave the fields blank

11. Click **Save**.

If you select the /News page and switch to **Preview** mode, you can see a list of all news pages placed under the **News** section.



This is an example of how content is structured in Kentico. 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 accessible to both people and search engines.

How it works

1. A visitor arrives on the **/News** page.
2. The **CMSRepeater** control placed on the page template checks if a news page is currently selected (based on the value of the **Clas**

sNames property).
3. The control finds out that the current page is a page (menu item), so it looks for all underlying news pages 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 web parts and controls have the **Path** property that specifies which content the component loads and displays. The following expressions are examples that you can use to select pages:

Path expression	Meaning
/%	All pages on the website.
/news/%	All pages under /News.
/news/news1	The News1 page.
./%	All items under the current page.
./logo	The Logo page under the current page.
./images/%	All pages under the Images page, which is a child of the current page.
../contacts/%	All pages under the Contacts page on the same content level as the current page.
/{0}/%	All pages under the page located on the first level of the current path. Example: If the currently selected page is: */news/news1* the system evaluates the expression as: */news/%*

Developing the Services page (ASPX)

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
2. Right-click the **CMSTemplates/MySite** folder in the Solution Explorer and click **Add -> Add New Item**.
3. Create a **Web Form** named ***LeftMenuRightText.aspx*** and check **Select master page**.
4. Click **Add** and choose the ***MyMaster.master*** page from the **CMSTemplates/MySite** folder.
5. Enter the following HTML layout code into the **<asp:Content>** element on the page:

```
<table style="width: 100%;">
 <tr style="vertical-align: top;">
  <td style="width: 20%;">

  </td>
  <td style="width: 80%;">

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

6. Drag the **CMSTreeMenu** control into the first table cell and the **CMSEditableRegion** control into the second cell.
7. Set the following properties for the controls:

CMSTreeMenu:

Property	Value	Description

Path	/{0}/%	Configures the tree menu to display pages 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 page.

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 Page tab of the Pages application in the Kentico administration interface.

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

```
using CMS.UIControls;
```

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

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

10. **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.

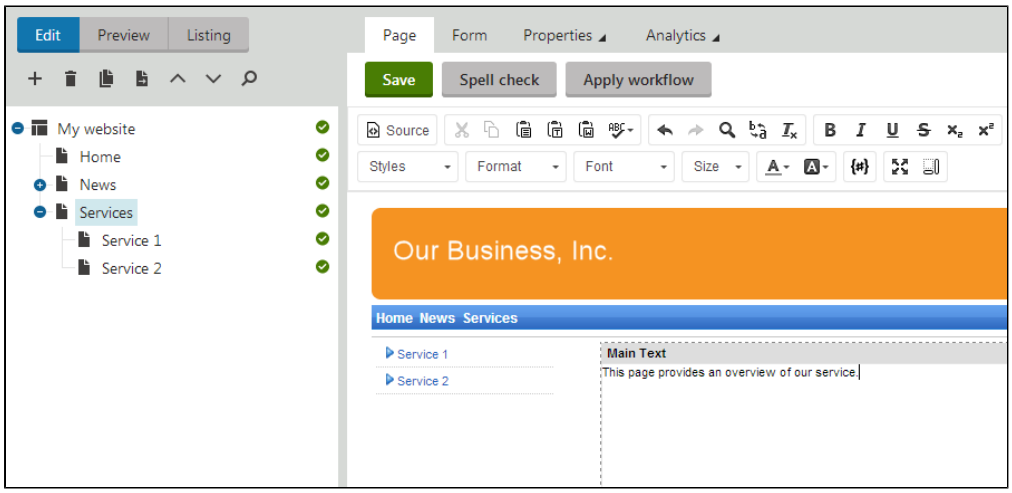
1. Switch to the Kentico administration interface in your browser.
2. Open the **Page templates** application.
3. Select the **My website** category.
4. Click **New template** and type ***Left menu with right text*** into the **Template display name** field.
5. Click **Save**.
6. Set the following values on the **General** tab:
   - **Template type**: ASPX page
   - **File name**: ~/CMSTemplates/MySite/LeftMenuRightText.aspx
7. Click **Save**.
8. 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. Open the **Pages** application.
2. Select the root of the content tree (**My website**).
3. Click **New** (+).
4. Choose the **Page (menu item)** page type.
5. 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.
6. 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. In the **Pages** application, select the **/Services** page.
2. Click **New** (+).
3. Choose the **Page (menu item)** page type.
4. 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.
5. 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 page, so you can enter unique content on each page.

Developing the Products section (ASPX)

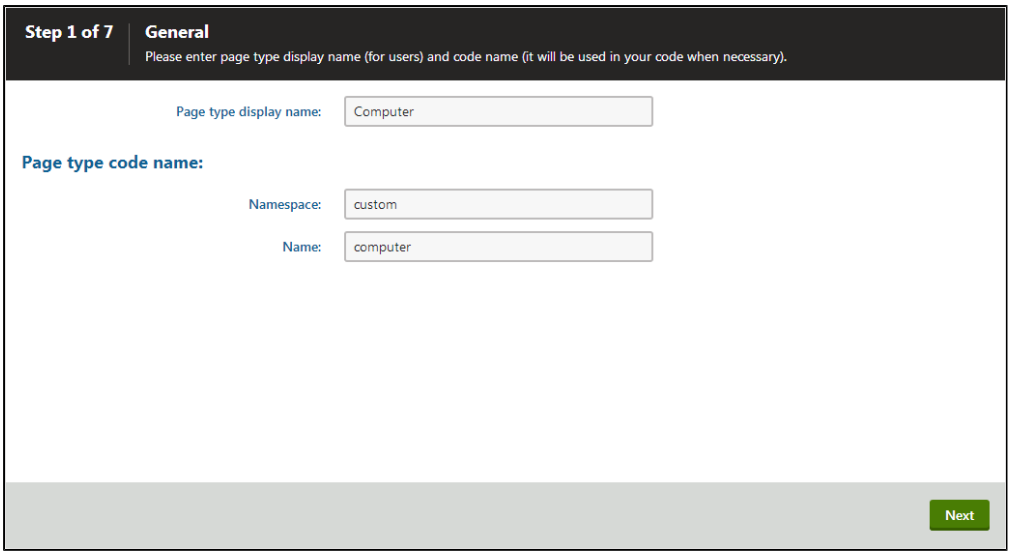
Now we will add a products section displaying a list of computers and their technical specifications. You will learn how to:

1. Create a new page type representing computer products
2. Write transformations
3. Create a page displaying a list of computer products

Defining a new page type (ASPX)

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

1. Open the **Page types** application.
2. Click **New page type**. This starts the **New page type** wizard.
3. Enter the following values in **Step 1**:
    - **Page type display name**: *Computer* (the system displays this name to users in the administration interface)
    - **Namespace**: *custom* (namespace to distinguish your page types from the default system types that use the cms namespace)
    - **Name**: *computer* (the identifier of the page type)



4. Click **Next**.
5. In **Step 2**, specify the name of the database table where the system stores the data of computer pages. You also need to enter the name of the table's primary key field. Leave the default values.
6. Click **Next**. The system creates a new database table for computer pages.
7. In **Step 3**, you need to define the fields of the page type (columns of the table). Click **New field** to create the following fields. For each field, enter the values, click **Save** and repeat the procedure until you have defined all the listed fields.

    - **Field name**: ComputerName
    - **Data type**: Text
    - **Size**: 200
    - **Required:** yes (checked)
    - **Field caption**: Computer name
    - **Form control**: Text box

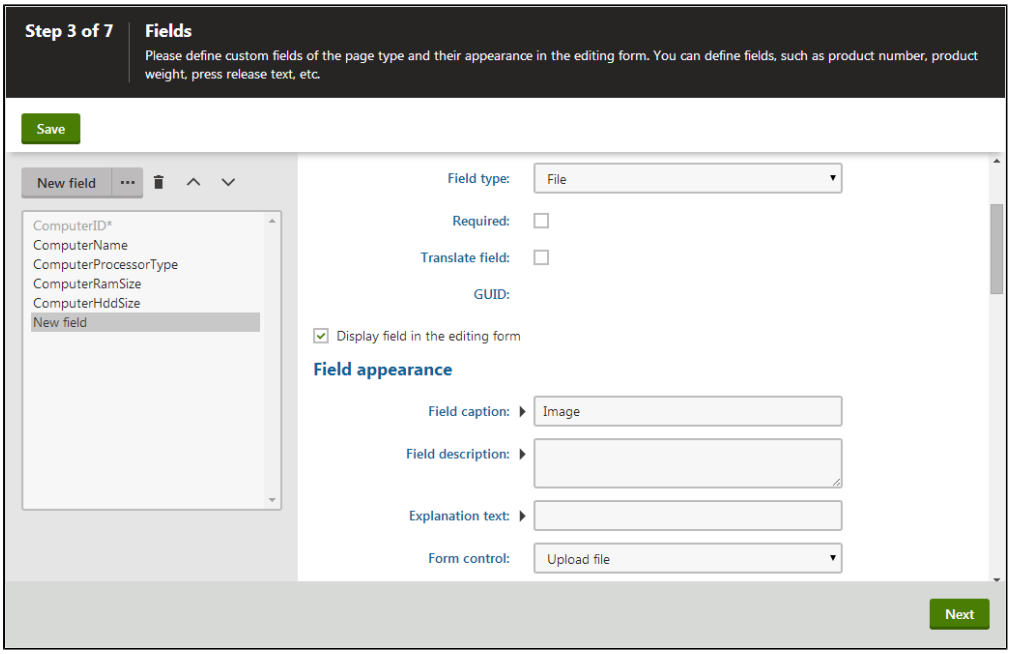
    - **Field name**: ComputerProcessorType
    - **Data type**: Text
    - **Size**: 200
    - **Field caption**: Processor type
    - **Form control**: Drop-down list
    - **Editing control settings -> Data source**: select **List of options** and enter the following items into the text area, one per line:

        ```
        Athlon;Athlon
        Pentium XEON;Pentium XEON
        Pentium Core 2 Duo;Pentium Core 2 Duo
        ```

    - **Field name**: ComputerRamSize
    - **Data type**: Integer number
    - **Field caption**: RAM (MB)
    - **Form control**: Text box

    - **Field name**: ComputerHddSize
    - **Data type**: Integer number
    - **Field caption**: HDD (GB)
    - **Form control**: Text box

    - **Field name**: ComputerImage
    - **Data type**: File
    - **Field caption**: Image
    - **Form control**: Upload file



8. Click **Next**.
9. In **Step 4**, choose the **ComputerName** field as the **Page name source**.

> This means that when a user creates a new computer page, the system automatically fills in the page name based on the **ComputerName** value. The page name appears in site navigation and in the content tree of the **Pages** application.

10. Click **Next**.
11. In **Step 5**, select the page types that will be supported as parents for computer pages in the content tree. Click **Add page types**, select the **Page (menu item)** page type and click **Select**. This means that users are only allowed to place computer pages under pages, not under articles, news items or other page types.
12. Click **Next**.
13. In **Step 6**, assign the page type to all websites where you wish to use it. Click **Add sites**, choose **My website** in the selection dialog and click **Select**.
14. Click **Next**.
15. Click **Finish** to complete the creation of the new page type.

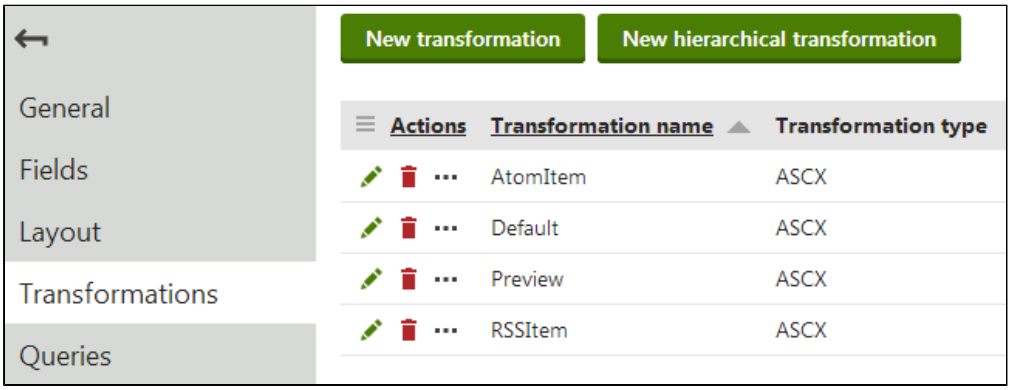
The wizard automatically creates the database table and several default transformations.

> **How does the system store page content?**
>
> The system stores page content and all related data in three database tables:
>
> - **CMS_Tree** (content tree structure)
> - **CMS_Document** (general page properties, metadata and editable region content)
> - A **dedicated page type table** - in this case ***CUSTOM_Computer*** (stores the values of the page type's specific fields)

Writing transformations (ASPX)

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

1. Open the **Page types** application.
2. Edit ( ) the **Computer** page type.
3. Switch to the **Transformations** tab.



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

4. 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>
```

> ASCX transformation code is similar to standard ItemTemplate elements that you may already be familiar with from from using ASP.NET Repeater or DataList controls. The transformation code combines HTML with ASP.NET commands and data binding expressions (Eval). You can also use built-in methods that simplify various tasks, such as **GetImage**. For more information about the available transformation methods, click the **Available transformation methods** link above the code editor.
>
> You will use the **Default** transformation for displaying the details of individual computer products.

5. Click **Save**.
6. 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>
```

7. Click **Save.**

Note the code used to create the link to specific pages. 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 page's **ComputerImage** field using the **GetImage** method. The sample code calls the method with a parameter that ensures automatic serverside 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 pages 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 press CTRL + SPACE to access a list of available page fields and related objects instead of typing them manually.

Creating the Products page (ASPX)

This page 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.
2. Right-click the **CMSTemplates/MySite** folder in the Solution Explorer and click **Add -> Add New Item**.
3. Create a **Web Form** named ***ProductList.aspx*** and check **Select master page**.
4. Click **Add** and choose the ***MyMaster.master*** page from the **CMSTemplates/MySite** folder.
5. Drag the following controls inside the **<asp:Content>** element of the product list page:
   - CMSBreadCrumbs
   - CMSDataList
6. Set the properties of the **CMSDataList** control according to the table below:

Property	Value	Description
ClassNames	custom.computer	Configures the datalist to display only pages of the custom.computer type.

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 ComputerName values.
TransformationName	custom.computer.preview	Assigns the transformation that the datalist uses to display the list of computer products.
SelectedItemTransformationName	custom.computer.default	When a user selects a specific computer page 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.

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

```
using CMS.UIControls;
```

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

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

9. Save the files.

Registering the page template

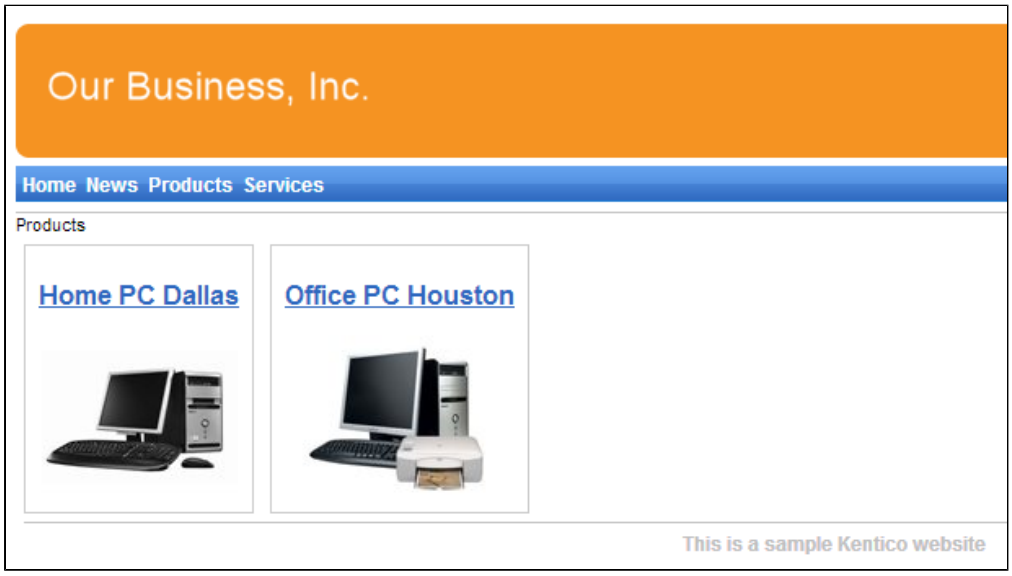
The source files of the products page are ready. Now you need to register the page template in Kentico.

1. Switch to the Kentico administration interface in your browser.
2. Open the **Page templates** application.
3. Select the **My website** category.
4. Click **New template** and type ***Product list*** into the **Template display name** field.
5. Click **Save**.
6. Set the following values on the **General** tab:
   - **Template type**: ASPX page
   - **File name**: ~/CMSTemplates/MySite/ProductList.aspx
7. Click **Save**.
8. Switch to the **Sites** tab and assign the page template to **My website**.

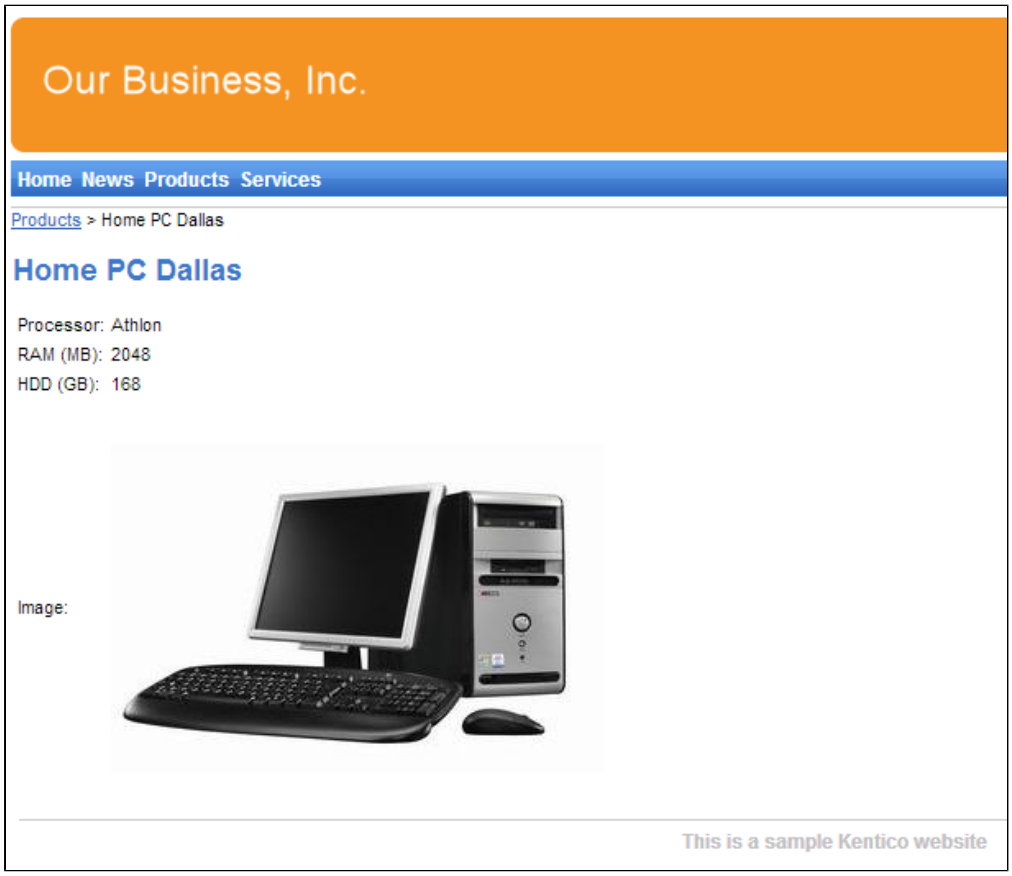
Adding the products section

1. Open the **Pages** application.
2. Select the root of the content tree (**My website**).
3. Click **New** (+).
4. Choose the **Page (menu item)** page type.
5. 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.
6. Click **Save** to create the page.
7. Select the **Products** page in the content tree.
8. Click **New** (+) and choose the **Computer** page type.
9. Fill in the computer page 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 images in the Sample web template - **SampleWebTemplate\Computer_Images**)
   - **Publish from/Publish to**: leave the values blank
10. 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 images in the Sample web template - **SampleWebTemplate\Computer_Images**)
    - **Publish from/Publish to**: leave the values blank
11. Click **Save**.

If you view the /Products page in **Preview** 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).



Developing the Search page (ASPX)

Kentico allows users to perform index-based searches through all page content, as well as other types of data. The following instructions describe how to add a basic search page to your website.

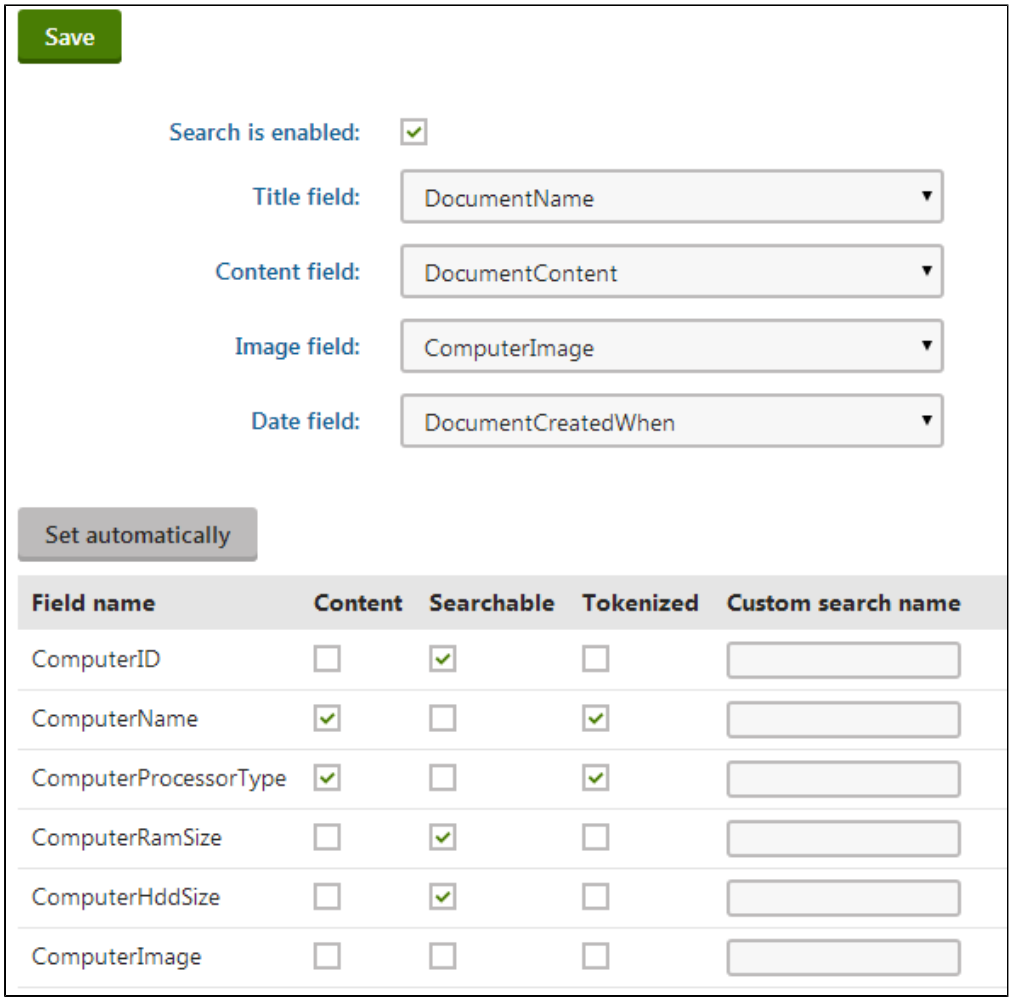
Configuring search fields for the Computer page type

First, set up the search options for the **Computer** page type that you created for the Products section.

1. Open the **Page types** application.
2. Edit ( ) the **Computer** page type.
3. Switch to the **Search fields** tab.

4. Select the **Search is enabled** check box.
5. Set the **Image field** to ***ComputerImage.***
6. Click **Save.**

Product pages are now searchable.

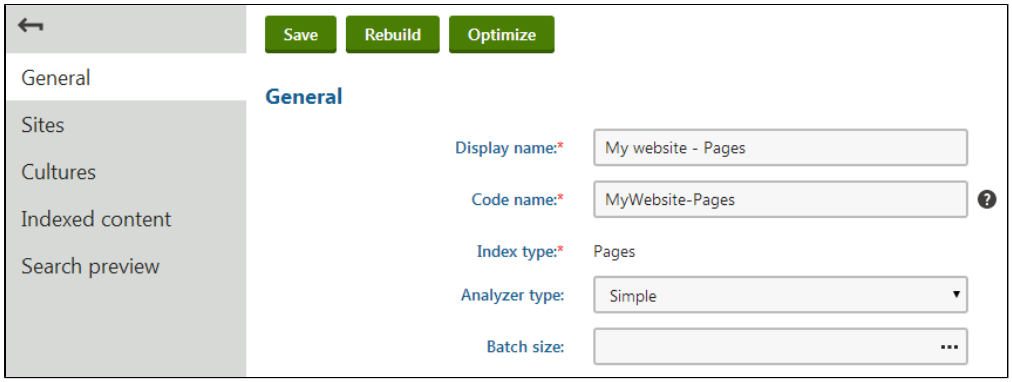


Creating a smart search index

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

1. Open the **Smart search** application.
2. Click **New index**.
3. Fill in the following details for the search index:
    - **Display name**: My website - Pages
    - **Index type**: Pages
    - **Analyzer type**: Standard
    - **Stop words**: (default)
4. Click **Save**. The index's editing interface opens.
5. Open the **Indexed content** tab and click **Add allowed content**.
6. Type */%* into the **Path** field and click **Save**. This ensures that the index includes all pages 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** section displays the current status of the index and other relevant information.



Preparing the ASPX source files for the search page

1. Edit your web project in Visual Studio
2. Right-click the **CMSTemplates/MySite** folder in the Solution Explorer and click **Add -> Add New Item**.
3. Create a **Web Form** named ***SearchPage.aspx*** and check **Select master page**.
4. Click **Add** and choose the ***MyMaster.master*** page from the **CMSTemplates/MySite** folder.
5. 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.

6. 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 - Pages** search index created in the previous section, which is assigned through the **Indexes** property (identified by the index code name).

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

```
using CMS.UIControls;
```

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

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

9. Save the search page files.

Registering the page template

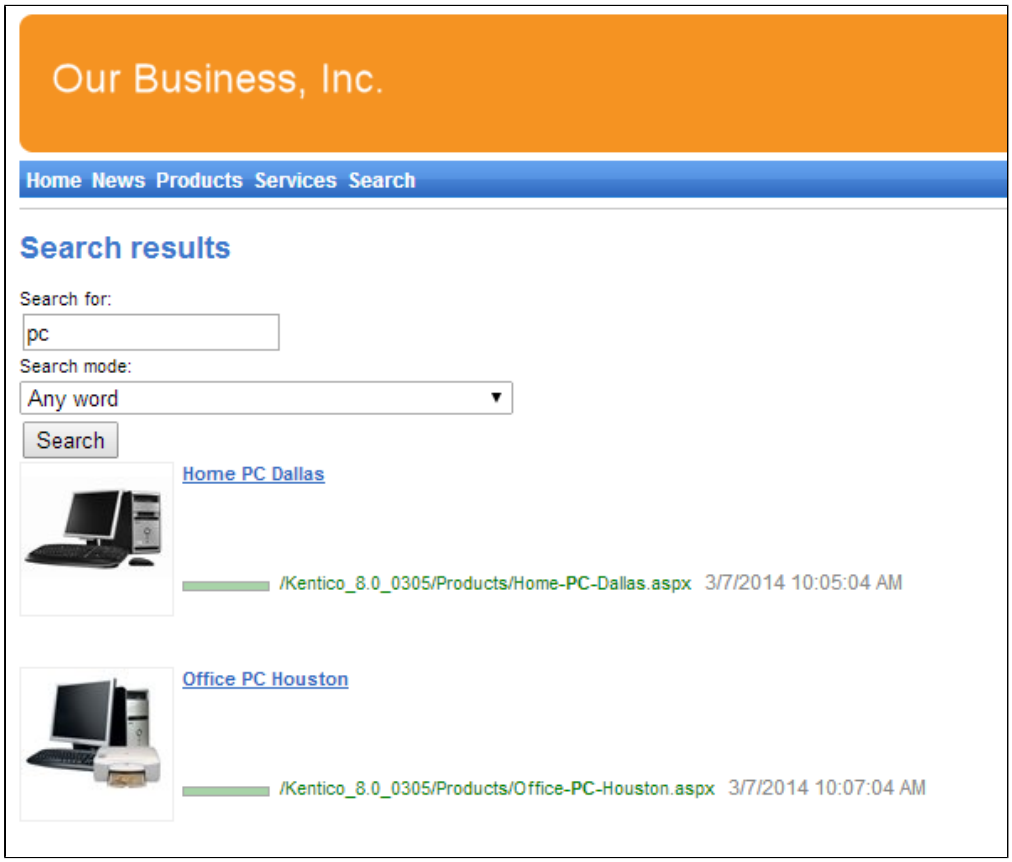
The source files of the search page are ready. Now you need to register the page template in Kentico.

1. Switch to the Kentico administration interface in your browser.
2. Open the **Page templates** application.
3. Select the **My website** category.
4. Click **New template** and type ***Search page*** into the **Template display name** field.
5. Click **Save**.
6. Set the following values on the **General** tab:
   - **Template type**: ASPX page
   - **File name**: ~/CMSTemplates/MySite/SearchPage.aspx
7. Click **Save**.
8. Switch to the **Sites** tab and assign the page template to **My website**.

Adding the search page

1. Open the **Pages** application.
2. Select the root of the content tree (**My website**).
3. Click **New** (+).
4. Choose the **Page (menu item)** page type.
5. 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.
6. Click **Save** to create the page.
7. Drag the Search page to the end of the content tree.

To try out the search functionality, view the /Search page in **Preview** 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 format of the search results

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

Adding a secured section for partners (ASPX)

Kentico provides a way to create secured site sections that can only be viewed by users who have a valid user name and password. This page describes how to create a logon web 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. The page reuses the template originally created for the website's Services page.

1. In the Kentico administration interface, open the **Pages** application.
2. Select the root of the content tree (**My website**).
3. Click **New** (+).
4. Choose the **Page (menu item)** page type.
5. 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.
6. Click **Save** to create the page.
7. Open the **Page** tab and type the following text into the editable region: *This is a secured page for partners.*
8. Click **Save**.
9. Open the **Properties -> Security** tab of the **Partners** page.
10. Select ***Yes*** for the **Requires authentication** property in the **Access** section
11. Click **Save**.

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

Creating the logon page

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
2. Right-click the **CMSTemplates/MySite** folder in the Solution Explorer and click **Add -> Add New Item**.
3. Create a **Web Form** named ***LogonPage.aspx*** and check **Select master page**.
4. Click **Add** and choose the ***MyMaster.master*** page from the **CMSTemplates/MySite** folder.
5. Enter the following HTML layout code into the **<asp:Content>** element on the page:

```
<table border="0" style="width: 100%">
 <tr style="vertical-align: 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>
```

6. Drag the following web parts (user controls) from the Solution Explorer into the left and right table cells respectively:
   - ~/CMSWebParts/Membership/Logon/LogonForm.ascx
   - ~/CMSWebParts/Membership/Registration/RegistrationForm.ascx
7. 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 email.
SendEmailFrom	no-reply@localhost.local	Sets the sender address for the password recovery emails.

RegistrationForm:

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

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

```
using CMS.UIControls;
```

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

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

10. Save the files.

Registering the logon page template

1. Switch to the Kentico administration interface in your browser.
2. Open the **Page templates** application.
3. Select the **My website** category.
4. Click **New template** and type ***Logon page*** into the **Template display name** field.
5. Click **Save**.
6. Set the following values on the **General** tab:
   - **Template type**: ASPX page
   - **File name**: ~/CMSTemplates/MySite/LogonPage.aspx
7. Click **Save**.
8. Switch to the **Sites** tab and assign the page template to **My website**.

Adding the logon page page

1. Open the **Pages** application.
2. Select the root of the content tree (**My website**).
3. Click **New** (+).
4. Choose the **Folder** page type.
5. Type in ***Special pages*** as the **Page name** and click **Save**.
6. Click **New** (+) again and select the **Page (menu item)** page type.
7. 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.
8. 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 pages of the Page (menu item) type. You can use folders to store pages that have a specific purpose on the website, but are not part of the 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 the Kentico administration interface. However, you can configure each website to use its own custom logon page.

1. Open the **Settings** application.
2. Select the **Security & Membership** category in the settings tree.
3. Select **My website** in the **Site** drop-down menu.
4. Clear the **Inherit from global settings** check 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.
5. Click **Save**.

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.
2. Edit the **MyMaster.master** master page (in the **CMSTemplates/MySite** folder).
3. Drag the following web parts (user controls) from the ***~/CMSWebParts/Membership/Logon/*** folder in the Solution Explorer and place them before the **CMSMenu** control (inside the ** ** element):
    - SignOutButton.ascx
    - CurrentUser.ascx
4. 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.

5. Save the changes.
6. Return to the Kentico administration interface in your browser and open the **CSS Stylesheets** application.
7. Edit the **My site stylesheet**.
8. Add the following class definitions to the stylesheet:

```
.CurrentUser
{
  color: white;
  padding-top: 4px;
}
.Right
{
  float: right;
  padding-right: 5px;
}
```

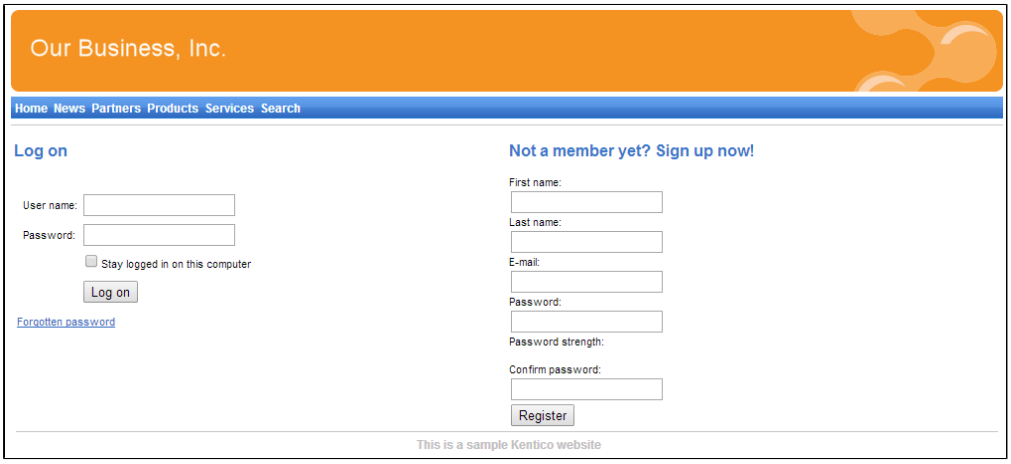
9. Click **Save.**

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. Open the user menu on the right of the Kentico administration interface header, and select **Sign Out**.
2. Click **Partners** in the main menu. The page is restricted, so the website redirects you to the logon page.



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

> Kentico 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.
>
> See Configuring permissions in the main documentation for more information.