Developer’s Manual

X3

Table of contents

Introduction

Extension 3 is a module developed for EPiServer CMS 5. This manual is intended for system and template developers with knowledge in ASP.NET 2.0 and EPiServer development and will only cover functionality specific for Extension. For instructions on EPiServer standard functionality we recommend you to read EPiServers Tech Notes available at www.episerver.com.

Extension

Extension allows people without programming skills to create and change page types, introducing a new way of working with EPiServer. Until now, editors had to turn to their system developer to add, remove och change the functions at the website. Extension will instead list them in a function library, allowing you to drag and drop them directly at the website. An intelligent permission system controls where you are allowed to drop them to maintain the structure and layout of the page. Everything is done in EPiServers user interface already familiar to you.

Getting Started

System requirements

- Extension files
- Put the Extension license under "/Dropit/License"
- Update the web.config with the configuration in the update.config.

Install manually with Extension Installer

- Extension files
- Put the Extension license under "/Dropit/License"
- Update the web.config with the configuration in the update.config.

Components

- Dropit.Extension.dll: Core component, contains the API classes.
- Dropit.Extension.UI.dll: GUI component contains all classes for rendering and edit mode.
- Dropit.Extension.Providers.dll: The Extension provider for storing the data.

Third party components

- Component Art, ComponentArt.Web.UI.dll
- JavaScript library Prototype
- JavaScript library Rico

Developer´s Guide

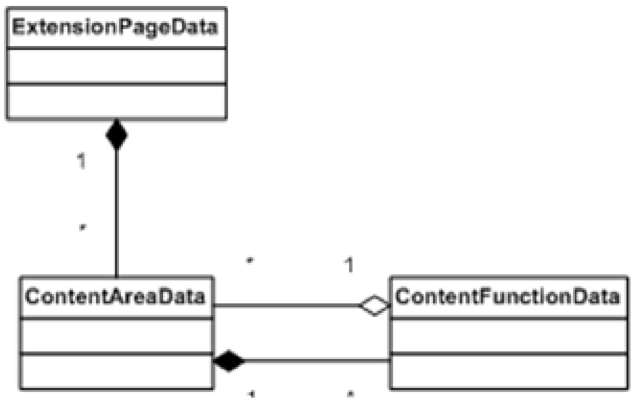
EPiServer vs. Extension

Extension 3 does not replace EPiServer. It gives the user the opportunity to create the content with more flexibility than traditional EPiServer. Let us make a simple comparison. In EPiServer, you work with the traditional asp.net way to create a web page by using a template page (.aspx) and units (user control, .ascx). Programmatically, you normally work with one object only, both from the page and its units. The CurrentPage object is a PageData and contains all page properties, such as Heading, MainIntro, MainBody. But with Extension, the CurrentPage object still remains for the page but each content function on the page has its own equivalent CurrentPage object, ContentFunctionData object and contains the content functions own properties.

Main API

The Page – Area – Function relation

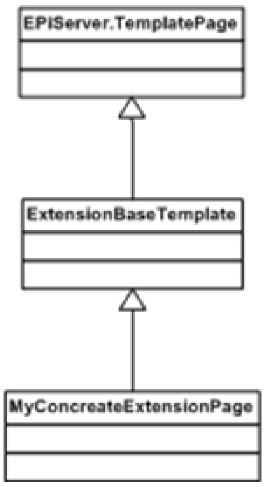
There are three main classes in the Extensions API: ExtensionPageData (equivalent to PageData), ContentAreaData for the Exetension Content Area and ContentFunctionData for the Extension content function. Below UML diagram describes the relation between these classes.



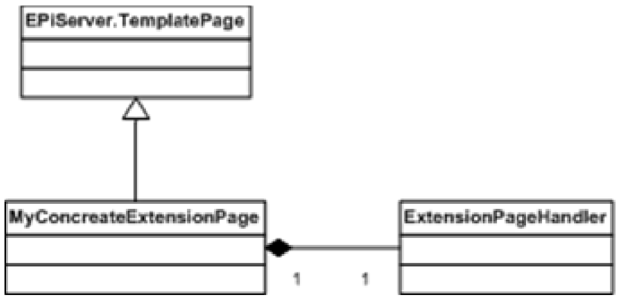
Create an Extension template

1. Create the base class

a) MyExtensionClass is either created as default or customized. The default one inherits the Dropit.Extension.Core.ExtensionBaseTemplate.



b) By choosing customized your Extension templates can inherit any base class, but need to implement an ExtensionPgaeHandler. It is recommended to create a base class when you choose this option for all Extension templates.



It is important to add the code to your code behind for the template class.

```
using Dropit.Extension.Core;

namespace  EPiServer.Templates
{
    public class MyExtensionTemplate : EPiServer.TemplatePage
    {
        // ExtensionPageHandler for Extension Functionality
        protected ExtensionPageHandler _extPageHandler;
        public MyExtensionTemplate(): base(0, 0)//extended by default
        {
            // create an instance of ExtensionPageHandler
            _extPageHandler = new ExtensionPageHandler();
        }
        protected override void OnInit(System.EventArgs e)
        {
            base.OnInit(e);
            // initialize ExtensionPageHandler instance
            // must always be placed at this position
            _extPageHandler.Initialize(this);
        }
    }
}
```

2. Define the Extension Areas

Use the Extension web control ExtensionContentArea (Dropit.Extension.Core) to define your template areas. Below example defines two Extension content areas, a MainArea and a RightArea.

```
<asp:Content ID=”MainContent”
    ContentPlaceHolderID=”MainBodyRegion”
    runat=”server”>
        <div id=”MainBody”>
            <Extension:ExtensionContentArea
                runat=”server”
                id=”MainArea”
             description=”The main content area,400px”
                />
        </div>
</asp:Content>
<asp:Content ID=”RightContent”
    ContentPlaceHolderID=”SecondaryBodyRegion”
    runat=”server”>
        <div id=”SecondaryBody”>
         <Extension:ExtensionContentArea
             runat=”server”
             id=”RightArea”
             description=”The right area for banners, 100px”
             />
        </div>
</asp:Content>
```

3. Register the template as an Extension page type

There are two ways of doing this.

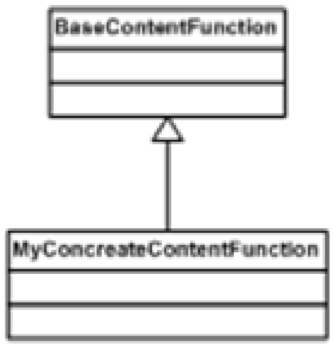
a) Register the template as an EPiServer page type exactly the same way you do with the standard EPiServer templates (Admin Page Type Create new page type). You can also add new properties to the page type if it’s required.

b) Go to Extension admin -> Page Types and follow the wizard to register the Extension template as an Extension page type.

Create an Extension function

Create simple content function

An Extension content function control is a user control and inherits BaseContent-Function (Dropit.Extension.Core).



Example on how to create a standard Extension content function with two properties - Heading and Mainbody.

```
<div>
    <h1>
        <Extension:Property
            ID="propHeading"
            PropertyName="Heading"
            runat="server" />
    </h1>
</div>
<div style="padding-top:10px;">
    <Extension:Property
            ID="propBody"
            PropertyName="MainBody"
            runat="server" />
</div>
```

Create a layout function

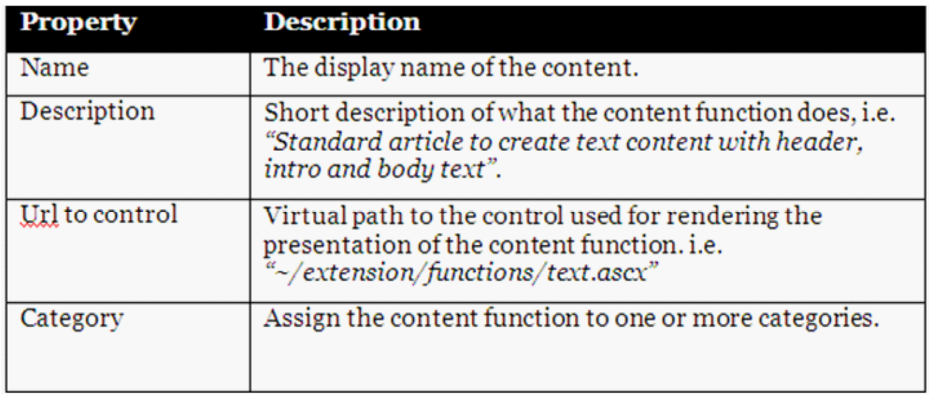
Add one or more content areas to a content function using the web control Content Area for defining layout content function (divider). Example on how to create a layout function with two equal columns.

```
<table style="width:100%;">
    <tr>
        <td style="width:50%;padding-right:10px;" valign="top">
            <Extension:ExtensionContentArea
                runat="server"
                id="LeftArea"
                />
        </td>
        <td  style="width:50%;padding-left:10px;" valign="top">
            <Extension:ExtensionContentArea
                runat="server"
                id="RightArea"
                />
        </td>
    </tr>
</table>
```

Register a content function

After creating a content function template, go to Extension Admin for registration: Extension Admin -> Content Functions.

1. Register the user control.



Property	Description
Name	The display name of the content.
Description	Short description of what the content function does, i.e. *"Standard article to create text content with header, intro and body text"*.
Url to control	Virtual path to the control used for rendering the presentation of the content function. i.e. *"~/extension/functions/text.ascx"*
Category	Assign the content function to one or more categories.

2. Add content function properties exactly as you do when defining an EPiServ er page type.
3. Set default access rights for newly created content function definitions can be changed in the settings.
4. Select the content areas that the content function are allowed to be dropped in.

Creating an Extension template page

An Extension template is a page with predefined structure and content. By selecting a template when creating a new page the user can maintain the layout and start editing with the content functions already in place. This reduces the risk of creating incorrect content in the wrong place and saves a lot of time for the editor who needs to create lots of pages with the same structure and layout.
Administrators can simply create them by clicking "Save as template" and give it a name, short description and an icon. The template will then be available to the editor as an option in the list of page types.

Working with global functions

A global function is an Extension content function with only one instance in the whole site but it still appears on several pages. If the global function instance is changed, it will affect all instances of this function and in all pages where it is used.
To create or edit a global function you need to be an Extension administrator. Create new global function from any content function instance on any Extension page by selecting the "Save as global function" option. All available global functions are listed under "Global function" section in the Toolbox.

Managing existing global functions

Extension administrators can edit the existing global function from the Administration tab in the Toolbox.

Working with dynamic content functions

A dynamic content function is a global function limited within a part of the page structure as a dynamic property. This dynamic property is applied only for the pages with the same page type as the root (the page where the dynamic property is set).

Using dynamic content functions

a. Create a dynamic property type of Extension Dynamic Content Function.
b. A dynamic content function is using a global function so you need to create a global function before setting the dynamic content function.
c. Setting the value to the content function:
- Select the content function
- Select the content area that the dynamic content will be applied to.
- Select the position for rendering the dynamic content (first or last).

Administrate Extension

Plug-in settings

1. Extension Data Container ID: Set the ID for the Extension Data container in the page tree.
2. Hide Extension Data Container: Show or hide the Extension Data Container from the page tree in edit mode.

Settings

1. "Extension administrators": Choose the user groups that should have admin istrator access rights:

a) Create Extension templates.
b) Create global functions.
c) Manage global functions and Extension templates from the Administration tab in the Toolbox.

2. "Root for Extension category": Select a root for the Extension categories from standard Category.

3. "Error handle level": Decide how Extension will handle errors. Except writ ing the messages to the log, the levels handle errors as following:

a) Default error handle (0): Extension forwards everything to the sender. Rec ommended during development phase.
b) Show friendly error message (1): Extension shows a more friendly error mes sage. Recommended during test phase.
c) Do nothing (2): Extension ignores all errors and tries to render anyway. Rec ommended during production.

4. Default access rights for new page types: Set the default access rights for newly created page types.

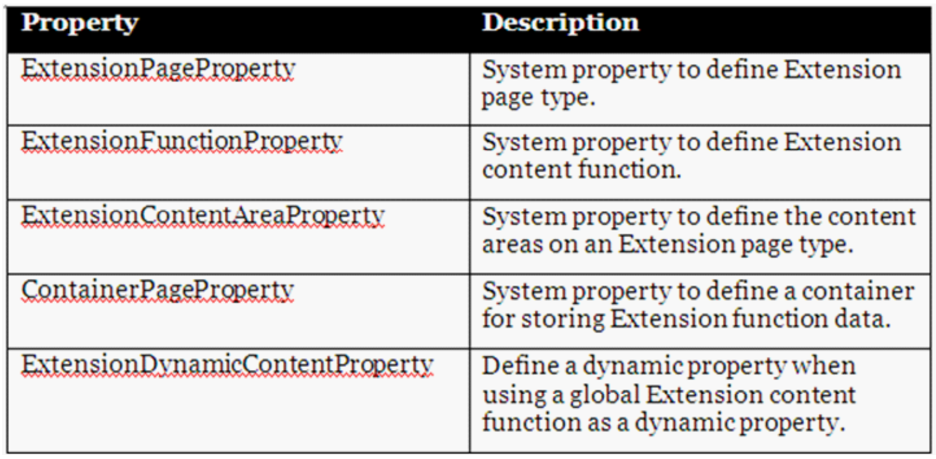
5. Default access rights for new content functions: Set the default access rights for new created content functions.

Create rules for content areas and content functions

The Extension administrator can set rules for what type of content is allowed in each content area and on each Extension page type. These rules can be set either in the rule management in the content function definition or in the page type definition. By default a content function can exist in all available content areas.

Extension specialized properties

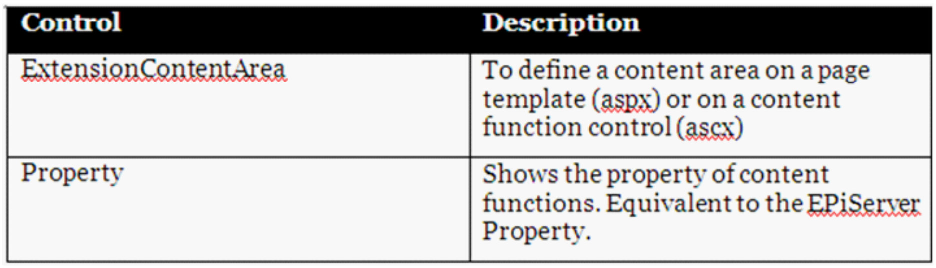
The Extensions specialized properties are located under Dropit.Extension.SpecializedProperties namespace



Property	Description
ExtensionPageProperty	System property to define Extension page type.
ExtensionFunctionProperty	System property to define Extension content function.
ExtensionContentAreaProperty	System property to define the content areas on an Extension page type.
ContainerPageProperty	System property to define a container for storing Extension function data.
ExtensionDynamicContentProperty	Define a dynamic property when using a global Extension content function as a dynamic property.

Extension web controls

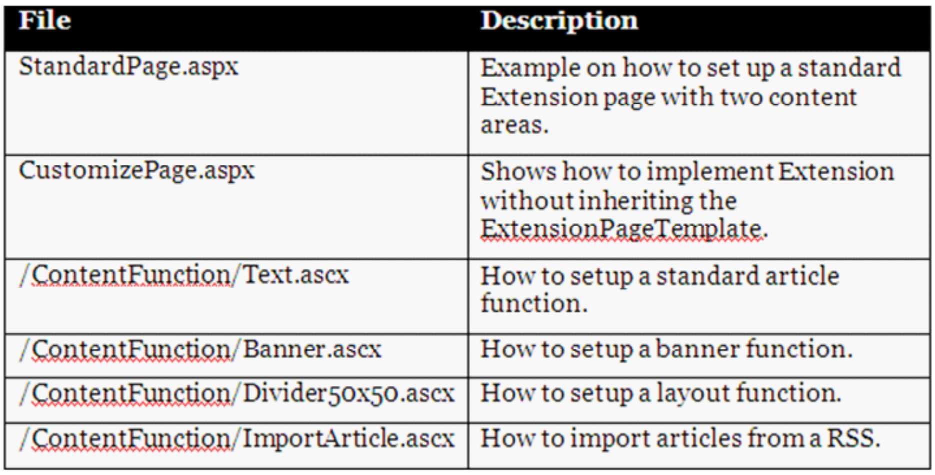
The Extension web controls are located under the Dropit.Extension.UI.WebControls namespace.



Control	Description
ExtensionContentArea	To define a content area on a page template (aspx) or on a content function control (ascx)
Property	Shows the property of content functions. Equivalent to the EPiServer Property.

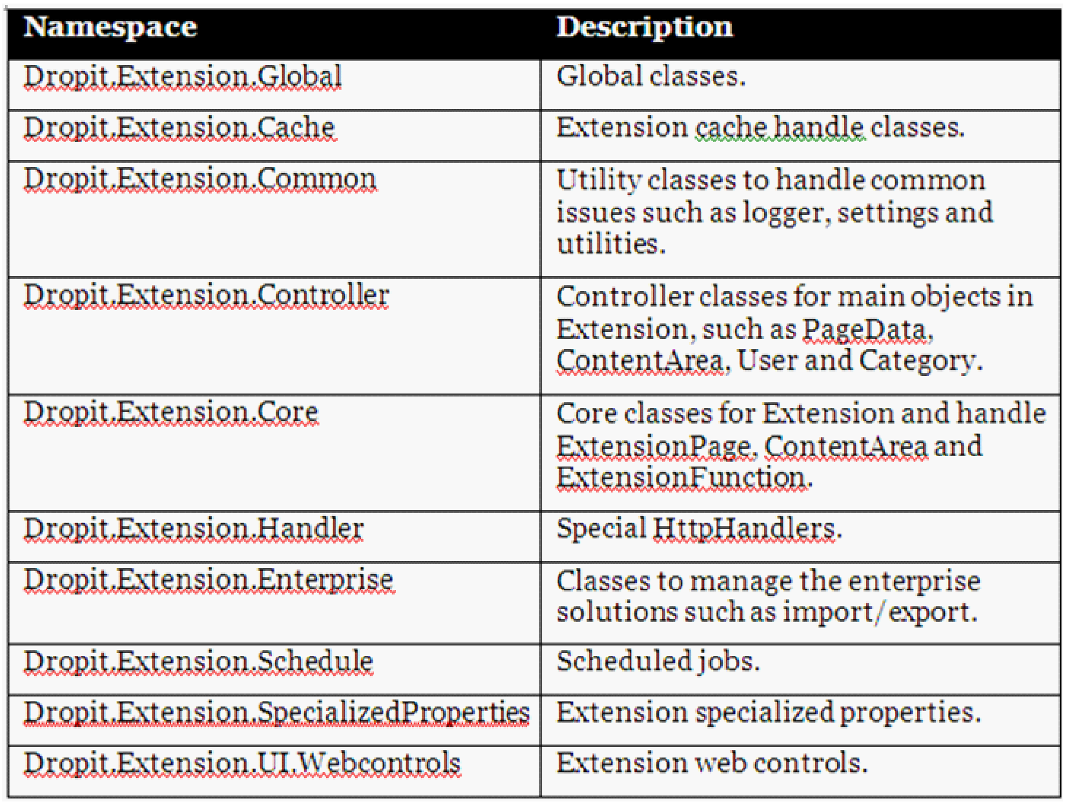
The sample project LoremIpsum

The sample project LoremIpsum contains examples on how to work with Extension for both beginners and advanced developers. Further examples will continuously be added to this project.



File	Description
StandardPage.aspx	Example on how to set up a standard Extension page with two content areas.
CustomizePage.aspx	Shows how to implement Extension without inheriting the ExtensionPageTemplate.
/ContentFunction/Text.ascx	How to setup a standard article function.
/ContentFunction/Banner.ascx	How to setup a banner function.
/ContentFunction/Divider50x50.ascx	How to setup a layout function.
/ContentFunction/ImportArticle.ascx	How to import articles from a RSS.

Namespaces



Namespace	Description
Dropit.Extension.Global	Global classes.
Dropit.Extension.Cache	Extension cache handle classes.
Dropit.Extension.Common	Utility classes to handle common issues such as logger, settings and utilities.
Dropit.Extension.Controller	Controller classes for main objects in Extension, such as PageData, ContentArea, User and Category.
Dropit.Extension.Core	Core classes for Extension and handle ExtensionPage, ContentArea and ExtensionFunction.
Dropit.Extension.Handler	Special HttpHandlers.
Dropit.Extension.Enterprise	Classes to manage the enterprise solutions such as import/export.
Dropit.Extension.Schedule	Scheduled jobs.
Dropit.Extension.SpecializedProperties	Extension specialized properties.
Dropit.Extension.UI.Webcontrols	Extension web controls.

Vocabulary

Content function	The Extension web controls are located under the Dropit.Extension.UI.WebControls namespace.
Layout function	Function used to split the content area in two or more columns.
Combined function	Combination of a content function and a layout function.
Global function	Content function defined on a global level.
Extension template	Template created in Extension, containing one or more functions.
Extension page type	Page type in Extension separate from EPiServer pages.
Toolbox	Dragable element to the right on the screen containing the different functions.
Toolbar	Element at the top of the screen containing a number of action buttons.
Clipboard	Area in the toolbox used for copying functions.
My favourites	Area in the toolbox used to store favourite functions.
Category	A function can belong to one or more categories.
Property	Property of a content function or combined function.
Edit On Page	Edit the Extension content directly from View mode using the Edit On Page mode
Function type	Functions belong to some of the following types: content/layout/combined/global
Access rights	Tells which person or group is able to create, change or delete a certain function
Access rules	Rules that set the access rights for users.
Content area	Area on an Extension template where you can create (drop) content functions.
Nested content area	Layout function inside another layout function.

More information

More information about Extension and other modules developed by Dropit AB is available at http://www.cmsapps.net.

Wiki

http://wiki.cmsapps.net is continuously filled with user information related to the CMSapps products. It is available to those of you who have purchased a license or received a demo account. Questions not covered in the wiki are directed to info@ cmsapps.net.

Blog

Get information about new releases and follow the development progress on upcoming products at http://cmsapps.wordpress.com/

Support

The module support team is available Swedish office hours between 9 am and 5 pm (GMT +01:00). Please note that the support service is intended for existing customers.

Phone: +46 10 522 64 80
E-mail: support@cmsapps.net