Installation & Configuration manual

ProcessMap

Version 3.5

Meriworks AB, Svensknabbevägen 17, 393 51 Kalmar, Sweden, www.meriworks.se

Installation & Administration Manual for Process Map version 3.5
Created September 1, 2006
Last edited September 16, 2016

Content

1. Introduction 3
1.1. Prerequisites 3
1.2. Manual conventions 3
1.3. Relation to other manuals 3

2. Requirements 4
2.1. Server 4
2.2. Client 4

3. Upgrade from version 2 5
3.1. Converting maps 5
3.2. Page property 5
3.3. Configuration 5

4. Installation for EPiServer 7.5 and below 6
4.1. Installation Step 1 - Information 6
4.2. Installation Step 2 – Choose web server 7
4.3. Installation Step 3 – Choose license file 8
4.4. Installation Step 4 - Finish 9
4.5. Finishing touches 10

5. Installation for EPiServer 8 or greater 11
5.1. Upgrading from ProcessMap prior to v3.4 11
5.2. Install ProcessMap via NuGet 12

6. EPiServer configuration 12
6.1. Add property to page type 13
6.2. Add property to template 14

7. Editor configuration 15

8. ProcessMap Property Settings for EPiServer CMS 8 21
8.1. Vertical line 22
8.2. Horizontal line 22

9. Windows SmartScreen warning 24

10. Security configuration modifications 25
10.1. Server settings 25
10.2. Client settings 25

11. Authentication modes and security settings 26
11.1. Security items 26
11.2. Authentication modes 26

12. Configuration 28
12.1. EPiServer CMS 8 28
12.2. ProcessMapExtension section 28

12.3. ConfigSections 28
12.4. Modules and Handlers 28
12.5. Application Settings 30
12.6. ProcessMapBrokenLinkReporter 31

13. Create new symbols 33
13.1. Symbol xml definition 33

14. Property Rules 36
14.1. Conditions 37

1. Introduction

Welcome!

ProcessMap is a plug-in for EPiServer. It is used to visualize processes and to connect the various steps of a process to e g documents, web pages or other sources of information. The ProcessMap plug-in is fully integrated into the EPiServer environment and can utilize standard EPi functions like search and version control. Process maps are easily created with the built in and easy to use drawing tool.

This manual is aimed towards an administrator or developer.

1.1. Prerequisites

The administrator will need a thorough understanding of EPiServer administration. The developer will need thorough understanding of xml and asp.net.

1.2. Manual conventions

Certain typographic conventions are used in this manual.

Running text is presented in the times font. Notes, tips and warnings are presented in bold.

Code is written in the courier typeface

```
print "Hello world"
```

Note! A note. Highlights important information.

Tips! A tip. Contains an advice or an easier way to do something.

Warning! A warning! Highlights a problem that might occur and how to avoid it.

1.3. Relation to other manuals

This manual is part of a series of two manuals. The other manual is ProcessMap: Editor manual. This manual is sufficient reading for an administrator or developer. Recommend reading is also the standard documentation for EPiServer.

2. Requirements

2.1. Server

ProcessMap works with the following versions of the .NET Framework and EPiServer:
.NET 3.5 or 4 and EPiServer 4, 5, 6, 7 or 8.

For EPiServer 7, the minimum version required is 7.0.859.1.

Make sure you have the installation package that suits your needs.

Note! There are different installation packages for EPiServer 7.1, 7.5 and 8.

Windows 2003 server and Windows 2008 server are supported web server operating systems.

2.1.1. EPiServer 7 and permanent links

If EPiServer versions prior to 7.1 is used, strict language routing may need to be disabled in order for ProcessMap links to successfully route. ProcessMap for EPiServer 7 has the following link format:

```
/link/[guid]?epslanguage=en
```

EPiServer versions prior to 7.1, with strict language routing, expects the language segment first like this:

```
/en/link/[guid]
```

For more information on strict language routing and how to disable it, see this article from EPiServer.

2.2. Client

The client that should view the process maps follows EPiServers recommendations for client hardware and software.

Clients that should edit the process maps follow EPiServers requirements for web editors in addition to the following:

Clients should have Internet Explorer 6.0 or later. Other browsers, such as FireFox or Chrome, may also work with a ClickOnce add-on.

Clients must have Microsoft .NET framework runtime of version 3.5 installed.

Vista clients must have the process map website added as a trusted site.

Windows XP, Windows Vista, Windows 7 and Windows 8 are supported client operating systems.

3. Upgrade from version 2

Upgrading from version 2 of ProcessMap will require a few manual steps.

3.1. Converting maps

Existing maps in ProcessMap 2 needs to be converted by a separate tool. The tool can be downloaded from www.meridium.se. The conversion tool cannot convert symbols so they need to be recreated using the new format and then the old symbols are replaced with the new once during the conversion. See the tools help or contact Meridium for more info.

3.2. Page property

ProcessMap 3 uses a different property type then version 2 so all page type files will need to be changed for old pages to continue working. Open the aspx files that are used by the page type and remove the registration of the ProcessMapExtension tag at the top and change the following line:

```
<ProcessMapExtension:ProcessViewer runat="server" />
```

It should look like this in ProcessMap 3:

```
<EPiServer:Property PropertyName="XmlDefinition"
DisplayMissingMessage="false" EnableViewState="false"
runat="server" />
```

3.3. Configuration

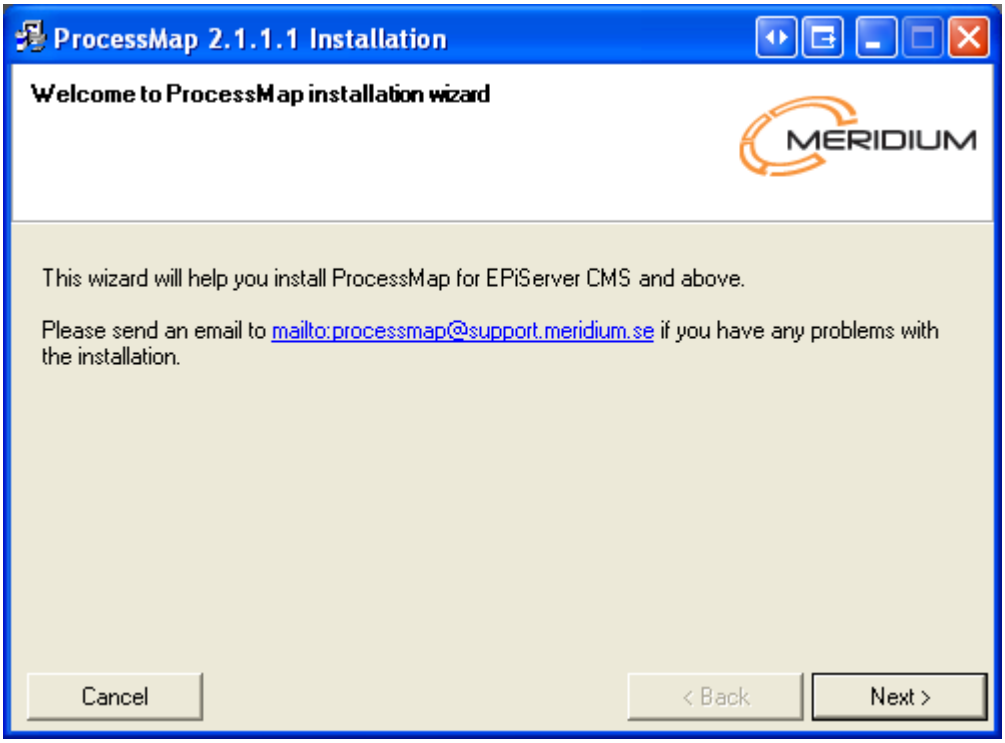
The old configuration file containing all settings will be deleted during the conversion. A new file with default settings will be created. Open the ProcessMap admin tool to add settings and add your custom symbols again. It is important that the symbols are added before starting to brows the maps otherwise the maps will be generated with a default symbol.

4. Installation for EPiServer 7.5 and below

ProcessMap is installed by running the installation application. The installation application copies the needed files, and makes the necessary modifications of the system configuration. The system configuration is modified by adding keys to the web.config file. A number of these keys are configurable as explained below.

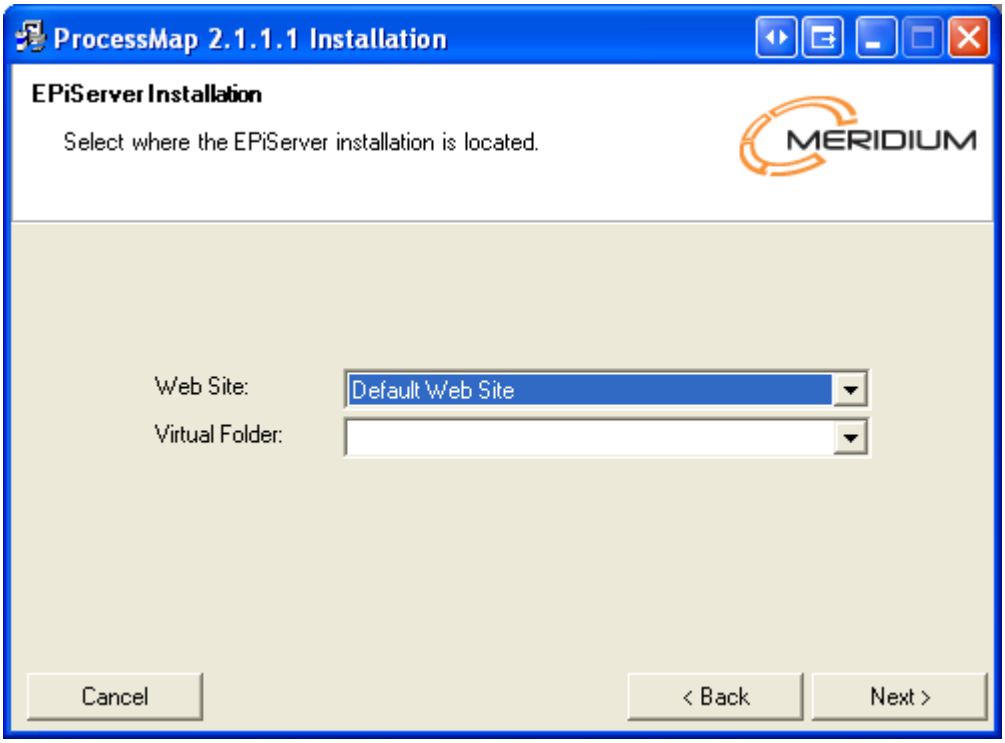
4.1. Installation Step 1 - Information

In the first step of the installation, prerequisites are stated. Make sure you have the right version of ProcessMap.



4.2. Installation Step 2 – Choose web server

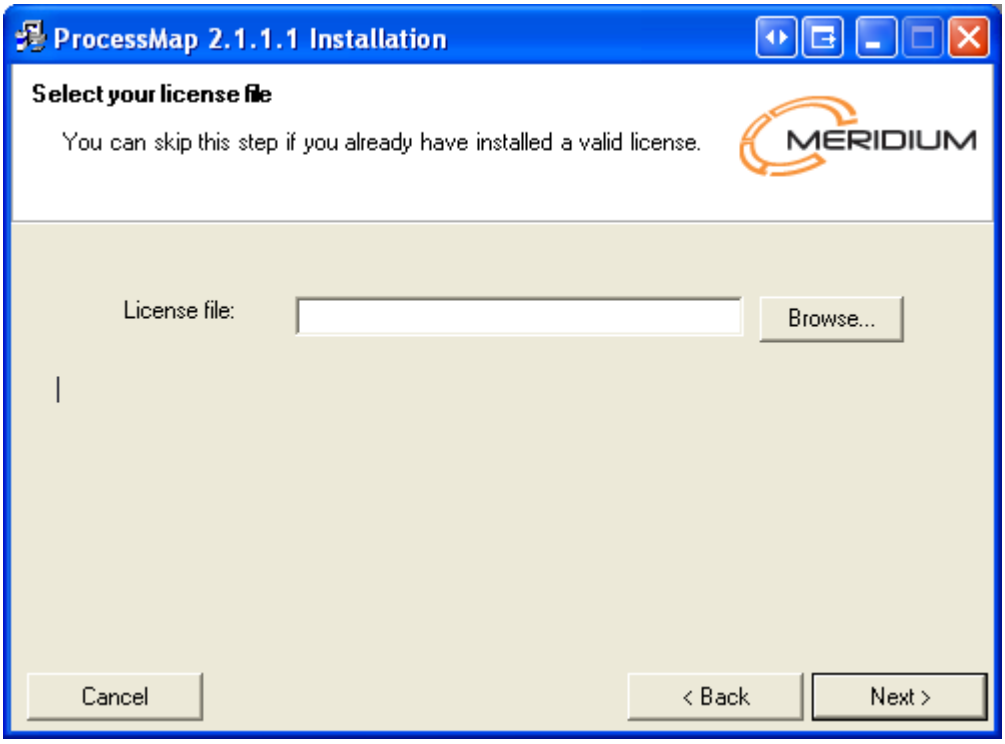
Select on which web site and virtual folder EPiServer is installed.



4.3. Installation Step 3 – Choose license file

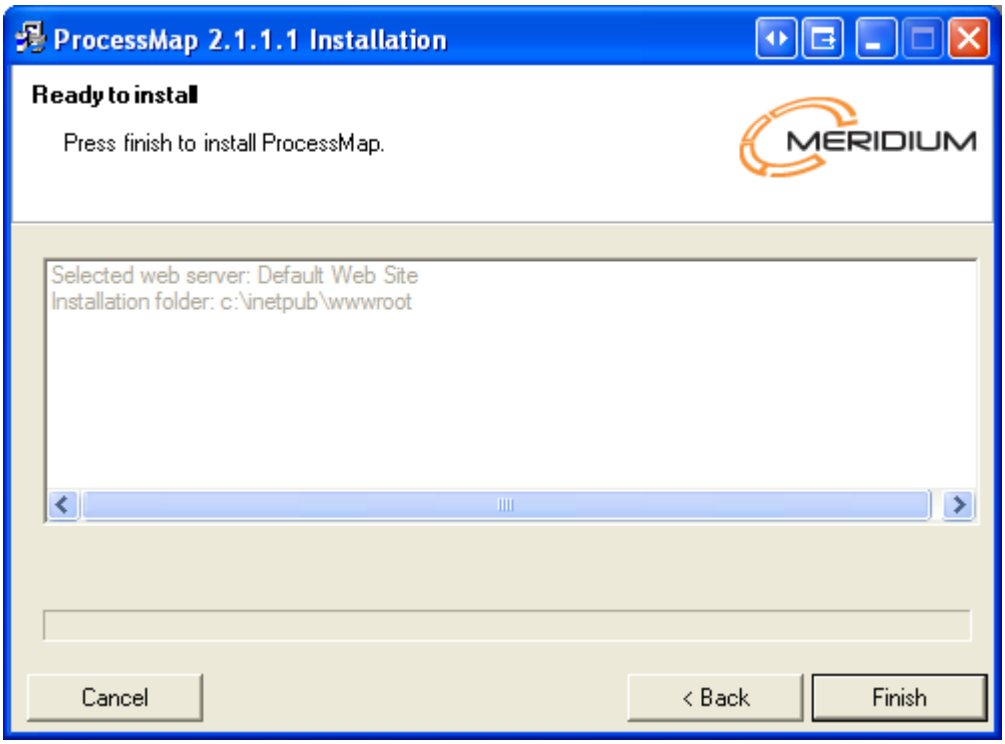
Browse to the folder where the license file you received with the installation package is located and select it. The license file is named ***meridiumLicense.config***. If you are upgrading the installation and already have a valid license, you can ignore this step and click next at once.

Tip! The license file can be found in the web root of the server (\Inetpub\mysite\) and is named meridiumLicense.config.

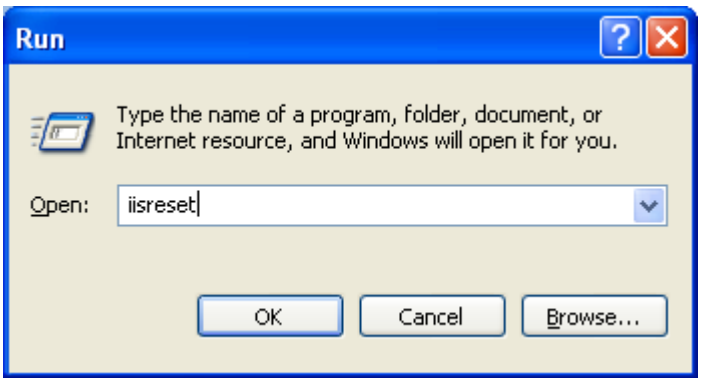


4.4. Installation Step 4 - Finish

Click on “Finish” to complete the installation.



When installation is completed it’s recommended to restart the web site with the iisreset command (or any similar method).



4.5. Finishing touches

4.5.1. Upgrade on EPiServer 7

If you are upgrading to ProcessMap for EPiServer 7 v 3.3 (or later), you might get some remaining configuration- and files in your installation, that are no longer used. This is because the MeridiumLinkEditor was removed for CMS 7 in ProcessMap v 3.3.
It is no problem leaving those files behind but if you want to clean them up, follow the instructions below.

Web.config

There is a configuration section in web.config named MeridiumLinkEditor that can be removed (example highlighted below).

```
<configuration>
  <configSections>
    <sectionGroup name="se.meridium">
      <section name="MeridiumLinkEditor" type="MeridiumLinkEditor.Config.Meridium
LinkEditorConfigurationSection,MeridiumLinkEditor, Version=1.0.4.0, Culture
=neutral, PublicKeyToken=01f4840270d48493"/>
    </sectionGroup>
  </configSections>
  <se.meridium>
    <MeridiumLinkEditor>
      <plugins>
        <add typeName="ProcessMap.EPiServer6, Version=3.2.1.0, Culture=neutral
, PublicKeyToken=01f4840270d48493" />
      </plugins>
    </MeridiumLinkEditor>
  </se.meridium>
</configuration>
```

Files

The following files can also be removed (relative path to the web root folder).

bin/MeridiumLinkEditor.dll
bin/sv/MeridiumLinkEditor.resources.dll
bin/no/MeridiumLinkEditor.resources.dll
Extensions/ProcessMapExtension/MeridiumLinkEditor

5. Installation for EPiServer 8 or greater

5.1. Upgrading from ProcessMap prior to v3.4

Note! If you already have an existing installation of ProcessMap that was installed using the method described in section 4, you need to remove the installation BEFORE installing the nuget package.
To upgrade from ProcessMap < 3.4 to ProcessMap >=3.4, follow the steps below.

5.1.1. Uninstall ProcessMap

To uninstall ProcessMap prior to version 3.4, but keeping the already created ProcessMaps, follow the instructions below.

Remove all ProcessMap assemblies from the bin folder and remove any ProcessMap references in the EPiServer project (project won't build until you installed the ProcessMap.EPiServer.UI nuget package later on).

Remove the following folders

- Plugins/ProcessMap/Client
- Plugins/ProcessMap/Style
- Plugins/ProcessMap/Script
- Plugins/ProcessMap/Images

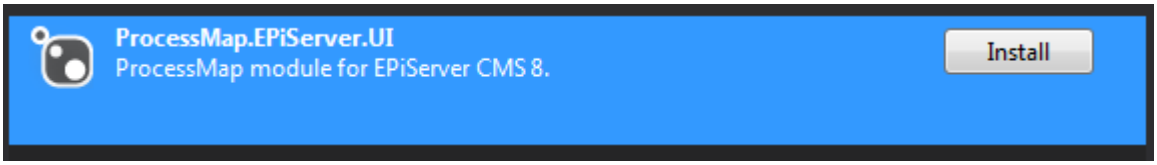
Remove all files in the Plugins/ProcessMap folder **[Observe! Keep the ProcessMapData and ProcessMapItems folders]**
In theory; all folders in the Plugins/ProcessMap folder except the ProcessMapData and ProcessMapItems folders should be removed. The ProcessMapData contains all edited ProcessMap files and ProcessMapItems contains the ProcessMap templates.

In Web.config, remove the following segments;

- Dependent assembly directives for ProcessMap assemblies.
- [Optional] Handler directives for ProcessMapImage.axd. (we supply a new URL for the ProcessMapImage.axd (~/Plugins/ProcessMap/ProcessMapImage.axd) that all generated html maps will use (from version 3.4) that is configured in the ~/plugins/processmap/web.config file). If you have existing html maps in the ProcessMapData folders, you need to delete those to use the new URL. You can keep the html files but then you also need to keep the handler directives or the Images in the link menus won't work.
- Remove all appSettings in the /configuration/se.meridium/ProcessMap/appSettings EXCEPT the XMLDataPath and TemplateFolder appSettings.
- [Optional] If you want to remove the complete /configuration/se.meridium/ProcessMap section, you need to move the XMLDataPath and TemplateFolder folders referred by the configuration, to the default location that is (from ProcessMap v3.4) ~/App_data/ProcessMap/ProcessMap_XMLDefinitions and ~/App_data/ProcessMap/ProcessMapItems respectively.

5.2. Install ProcessMap via NuGet

Starting with EPiServer CMS 8, ProcessMap installation has been simplified and is now installed as a NuGet package into your existing web project. The package is located on the official nuget.org feed and is called **ProcessMap.EPiServer.UI**.



Locate the package with your NuGet package manager and add it to your web project by clicking Install. The package will install all the required files and perform required configuration changes to your project.

6. EPiServer configuration

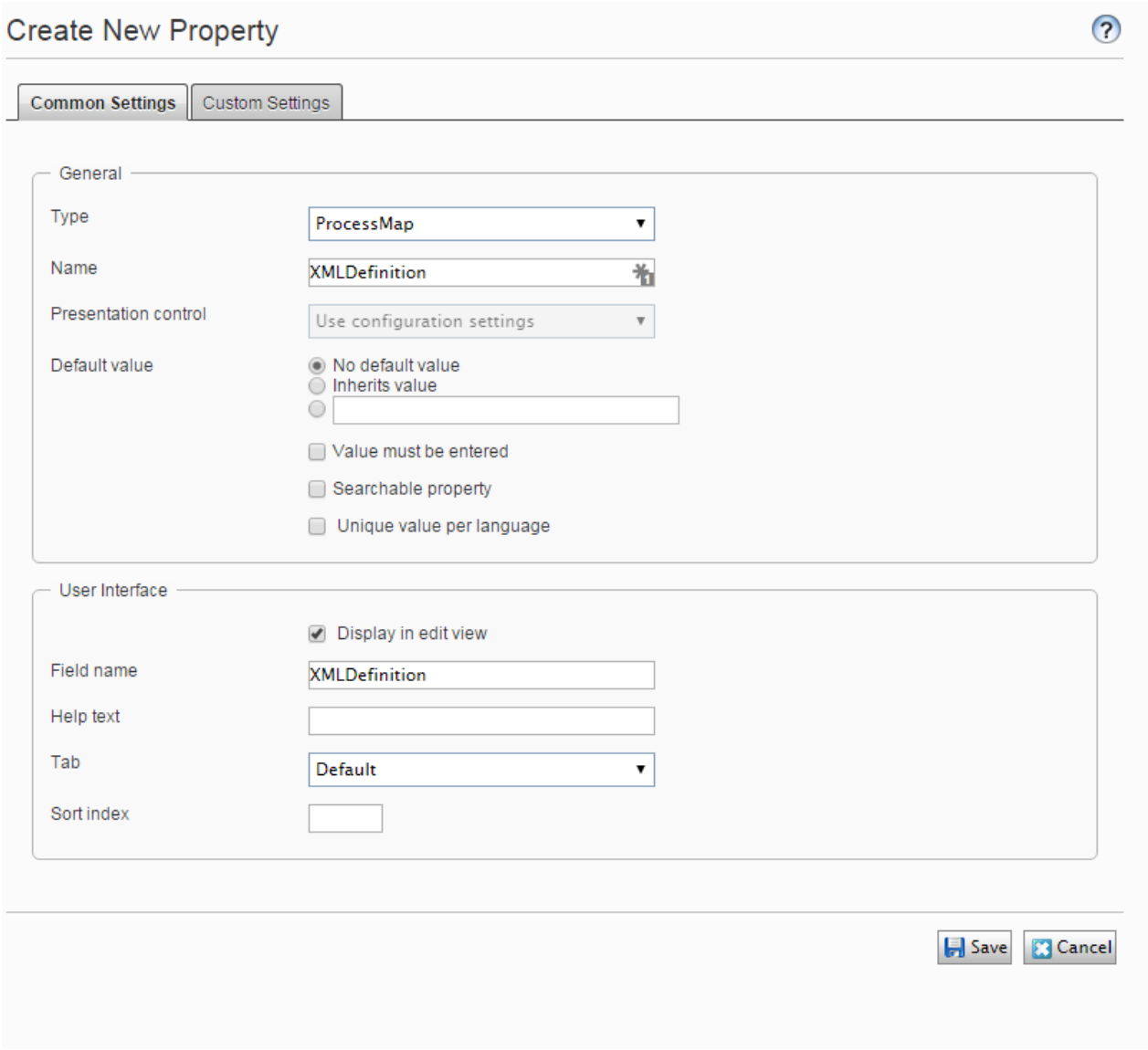
In order to use ProcessMap on a page in EPiServer you need to create a new page type that contains the *XMLDefinition* property that was installed during the installation process.
The installation process will create a ProcessMap PageType and install a file named PageProcessMap.aspx in the templates folder. This file is to be used as an example for creating your own templates.

Note! The example template is made for the Alloy Tech Sample Site. If another set of templates are used changes must be made accordingly in the example page template to ensure that the process maps are shown correctly.

Warning! Any changes made to the file PageProcessMap.aspx will be replaced when ProcessMap is reinstalled or updated.

6.1. Add property to page type

Click on *Add property* on the page type that you want to use. Set *Name* to XMLDefinition and choose ProcessMap from the *Type* dropdown box.



6.2. Add property to template

Add following line to the register section of the template.

```
<%@ Register TagPrefix="EPiServer"
Namespace="EPiServer.WebControls" Assembly="EPiServer" %>
```

Add the following line where you want to display the process map.

```
<EPiServer:Property PropertyName="XmlDefinition" runat="server" />
```

The maps requires a head tag on the template with the attribute runat=”server”. If you by some reason do not want to add this attribute, the following html must be added inside the head tag. (The stylesheet is used for styling the popup menus that are displayed when a symbol with links is clicked. This stylesheet can of course be customized.)

```
<link rel="stylesheet" type="text/css"
href="/Extensions/ProcessMapExtension/style/pm-style.css"/>
<script type=""text/javascript"
src="/Extensions/ProcessMapExtension/script/pm-property.js"
"></script>
```

6.2.1. MVC

The ProcessMap properties can also be used with MVC. A display template for the property is included with the installation. First include the needed namespaces.

```
using ProcessMap.EPiServer.Property;
using ProcessMap.EPiServer.Common;
```

Then declare your property in the model, the type should be ProcessMapDataType and decorated with the BackingTypeAttribute pointing to the PropertyProcessMap type.

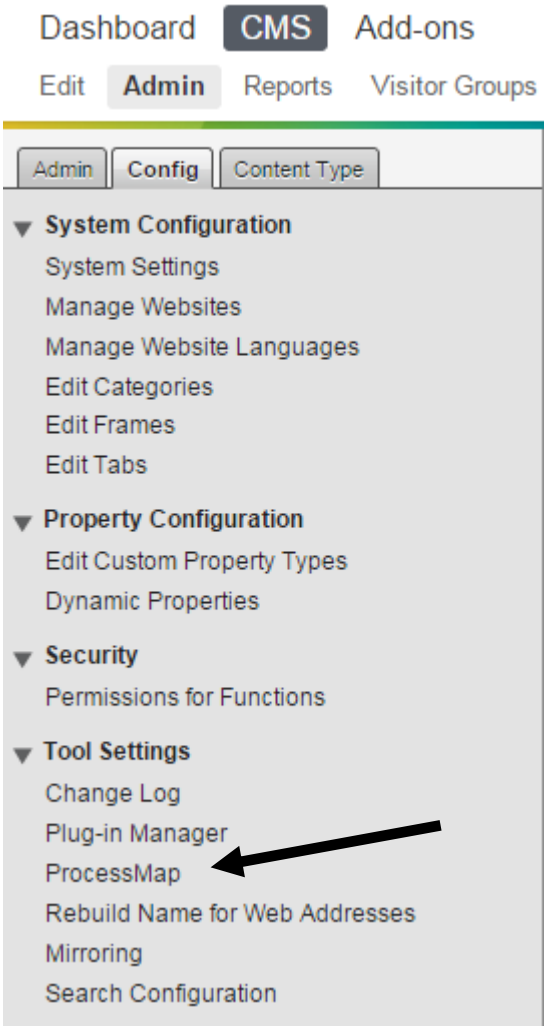
```
[Display(GroupName = SystemTabNames.Content, Order = 330)]
[BackingType(typeof(PropertyProcessMap))]
public virtual ProcessMapDataType MyProcessMap { get; set; }
```

Then render it in the view with the PropertyFor helper.

```
@Html.PropertyFor(x => x.MyProcessMap)
```

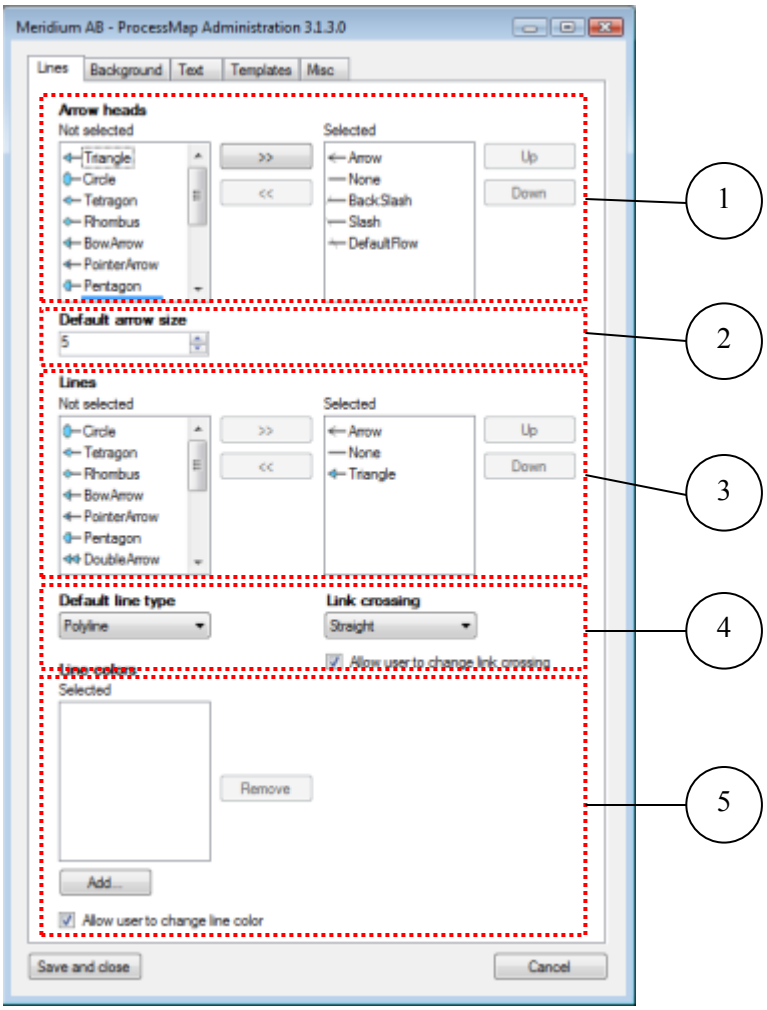
7. Editor configuration

It is possible to configure the behavior of the ProcessMap editor. All settings are stored in configuration.xml which can be found in the ProcessMap_XMLDefinitions folder. This file will be created the first time ProcessMap runs and is not included in the installation package. The file can either be changed manually or by using an application. The application is run from admin mode in EPiServer under “Tool Settings”. Look for the link “ProcessMap”.

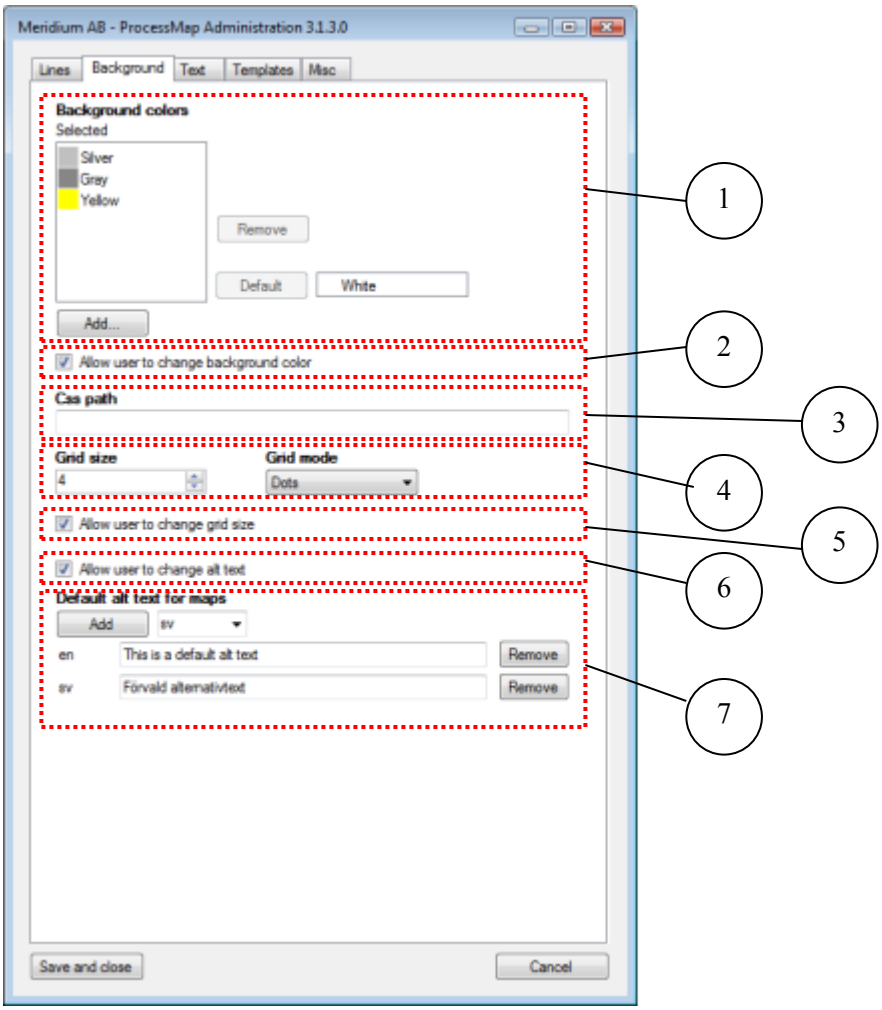


Warning! The web application needs to be restarted before the changes made to the configuration file takes effect.

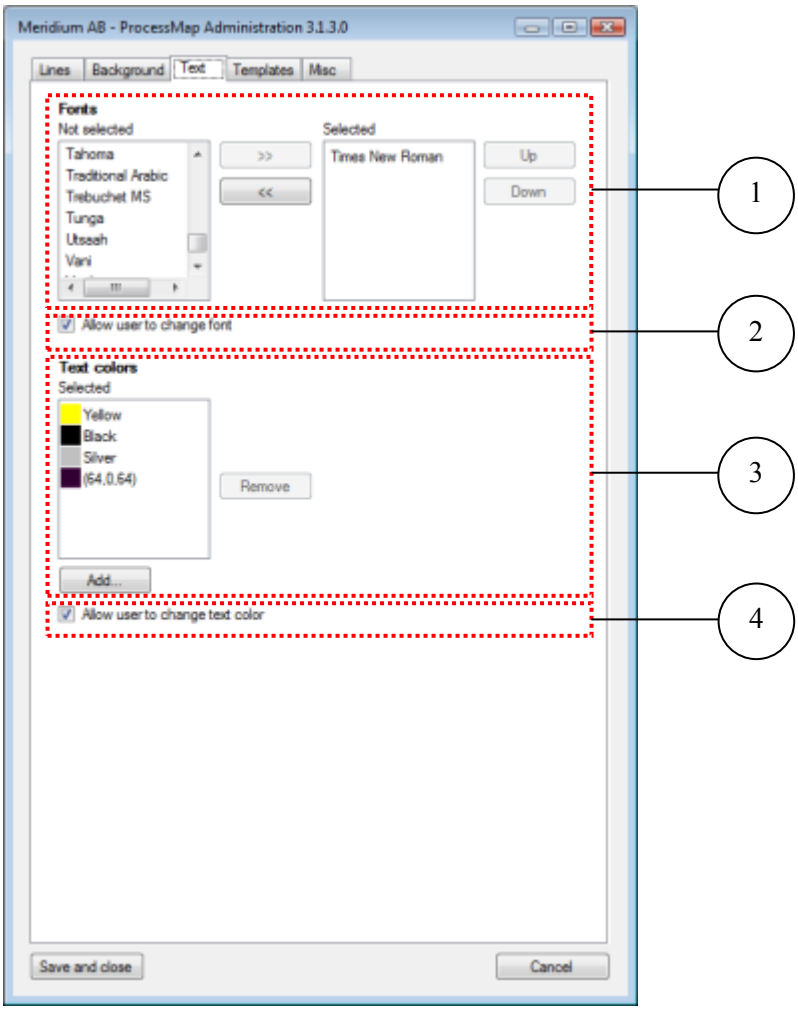
Below are description of each setting and the name of the element in the configuration xml file.



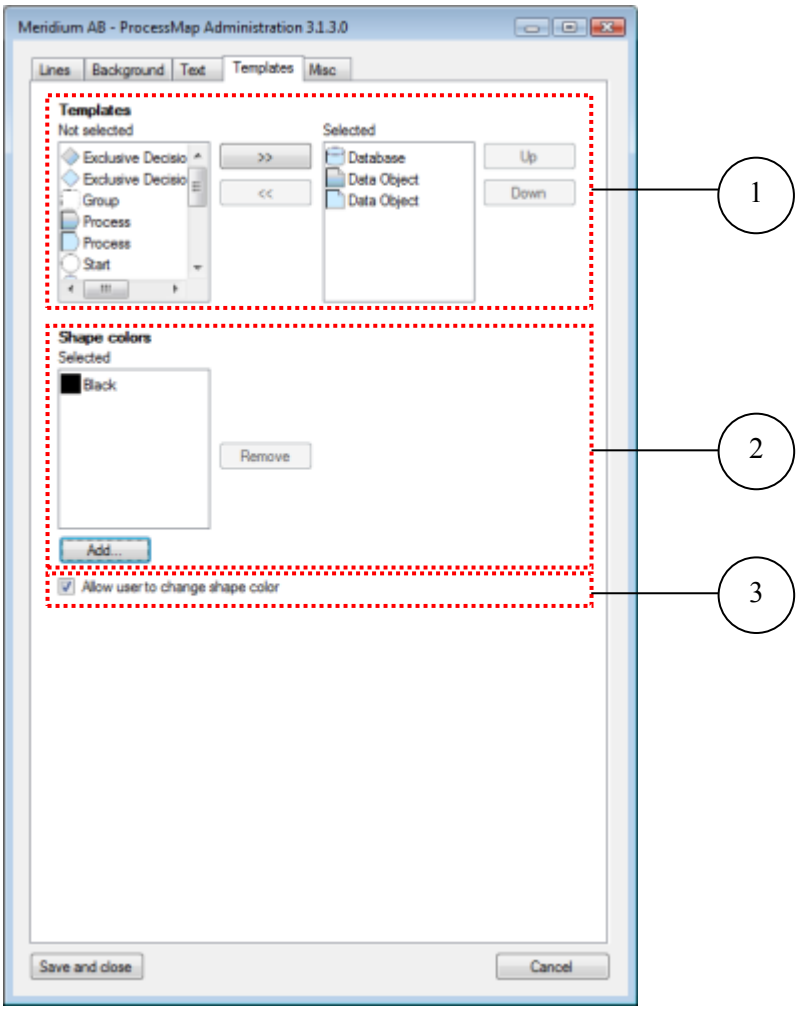
Name	Description
1. ArrowHeads	Controls which arrow heads that can be selected for a connector line. If omitted all arrow head will be available.
2. ArrowSize	The default size on arrows.
3. SelectedLines	This setting controls which connector lines that are available in the ProcessMap editor. See ArrowHeads.
4. DefaultLineType LinkCrossing ChangeLinkCrossing	Default line type. Polyline, Bezier or Cascading. Default connector line crossing. Straight, Arcs or Cut. If changing line crossing is allowed.
5. LineColors ChangeLineColor	Controls which colors that can be used for connector lines. See BackgroundColors. If omitted all colors will be available. If changing line color is allowed.



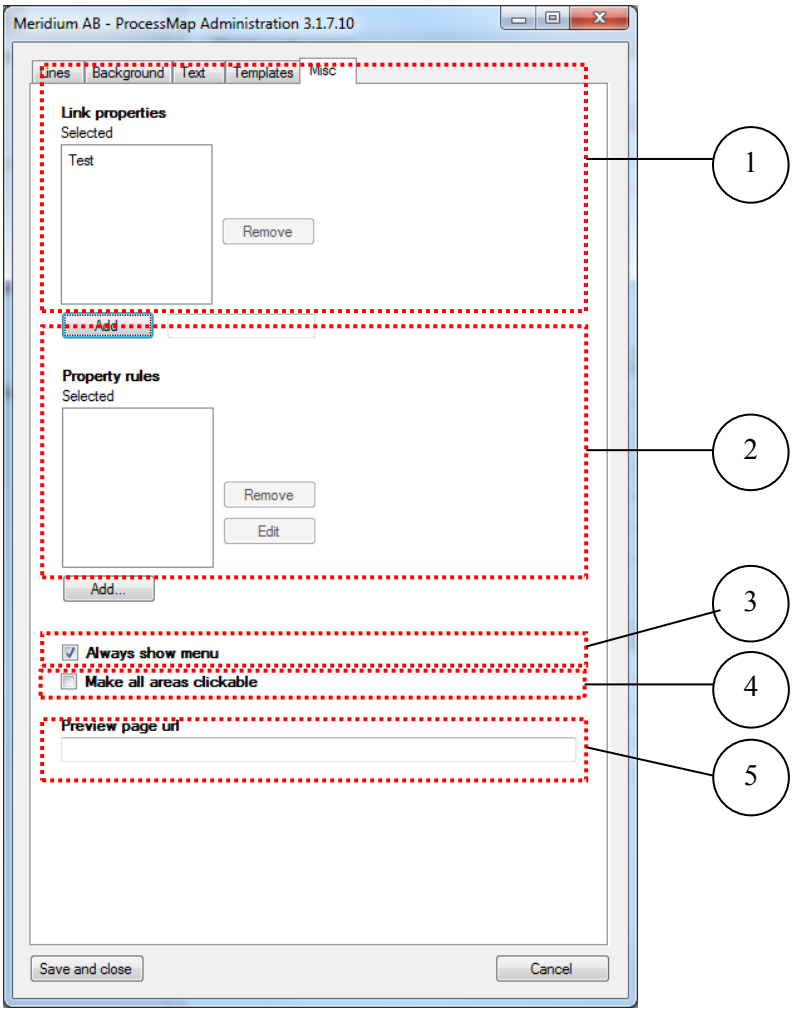
Name	Description
1. BackgroundColors DefaultBackgroundColor	Controls which colors that can be used as background. Colors can be specified as RGB, ARGB or by name. Specifies the diagrams default background.
2. ChangeBackgroundColor	If changing background color is allowed.
3. CssPath	A custom css file that will be used when displaying the ProcessMaps.
4. GridSize GridMode	Sets the grid size. Grid mode. None, Hidden, Dots or Lines.
5. ChangeGrid	If changing grid size and grid mode in the editor is allowed.
6. ChangeAltText	If editors should be allowed to change alt-text.
7. AltTextDictionary	A dictionary containing the default alt-text in different languages.



Name	Description
1. SelectedFonts	Which fonts that should be available in the editor. Make sure that the selected fonts exists both on the server and on the client.
2. ChangeFont	If changing font is allowed.
3. TextColors	Which colors that should be available for text in the editor.
4. ChangeTextColor	If changing text color is allowed.



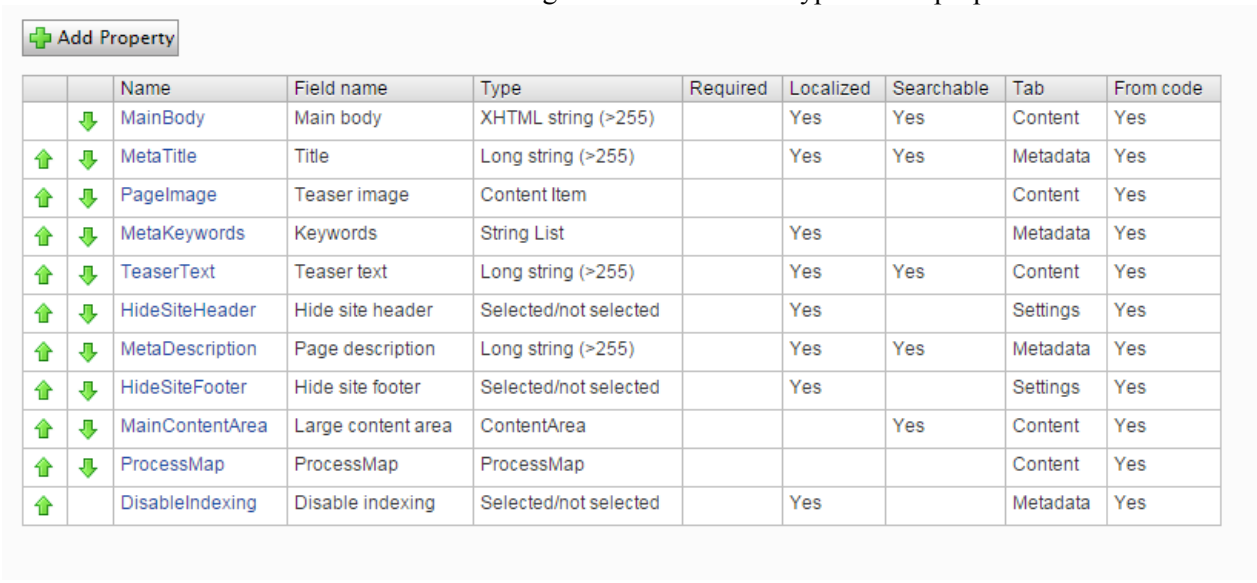
Name	Description
1. SelectedTemplates	This setting controls which symbols that are available in the ProcessMap editor.
2. ShapeColors	Which colors that should be available for symbols in the editor. See BackgroundColors.
3. ChangeShapeColor	Which colors that should be available for text in the editor.



Name	Description
1. LinkProperties	Defines what properties that can be selected for a link.
2. PropertyRules	Rules for the link properties. See chapter 14.
3. MenuAlwaysShow	If the link dropdown should be displayed even if there is only one link.
4. Make all areas clickable	If symbols should be considered links. This is required in order to use events in the SharePoint version.
5. PreviewPageUrl	The page specified is used for previewing processmaps in SharePoint. Only used in the SharePoint version!

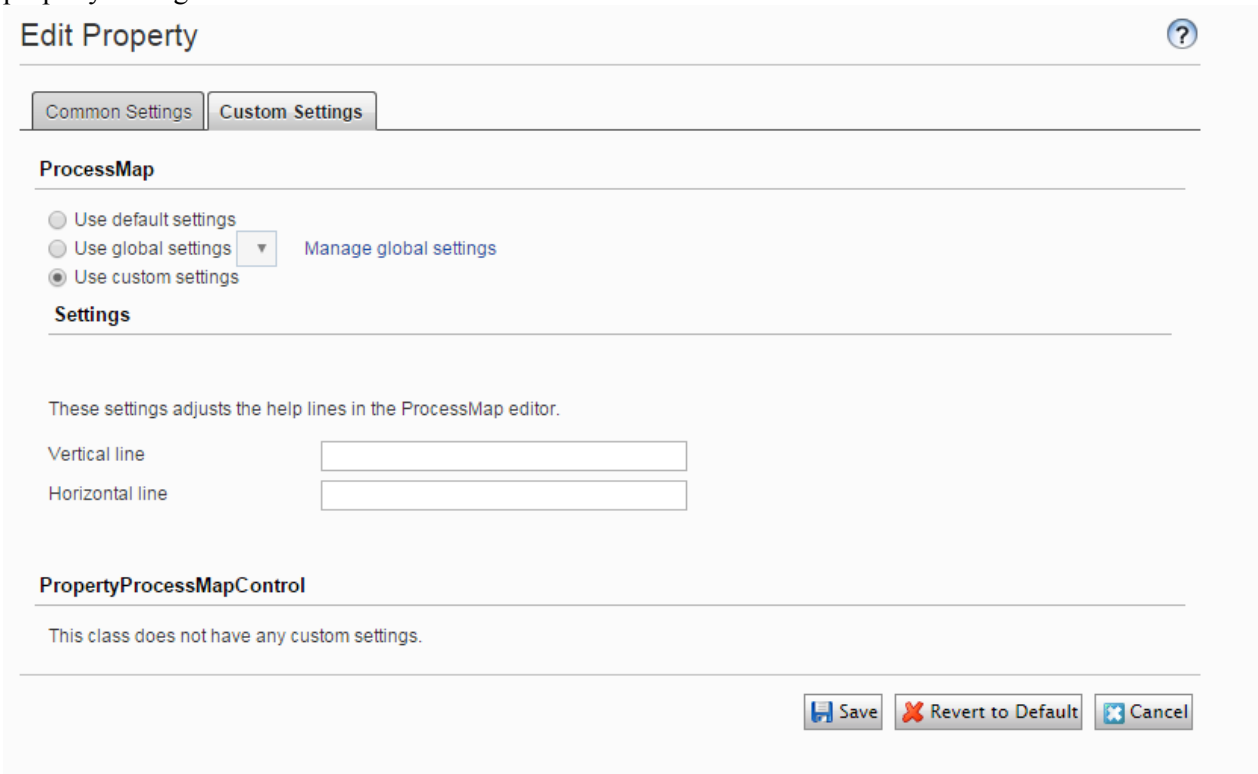
8. ProcessMap Property Settings for EPiServer CMS 8

Starting with EPiServer CMS 8 you can set property settings on ProcessMap properties. The settings that can be set are Vertical line and Horizontal line. This is set on a property level by going to the Admin section of EPiServer and locating the current content type and its properties.



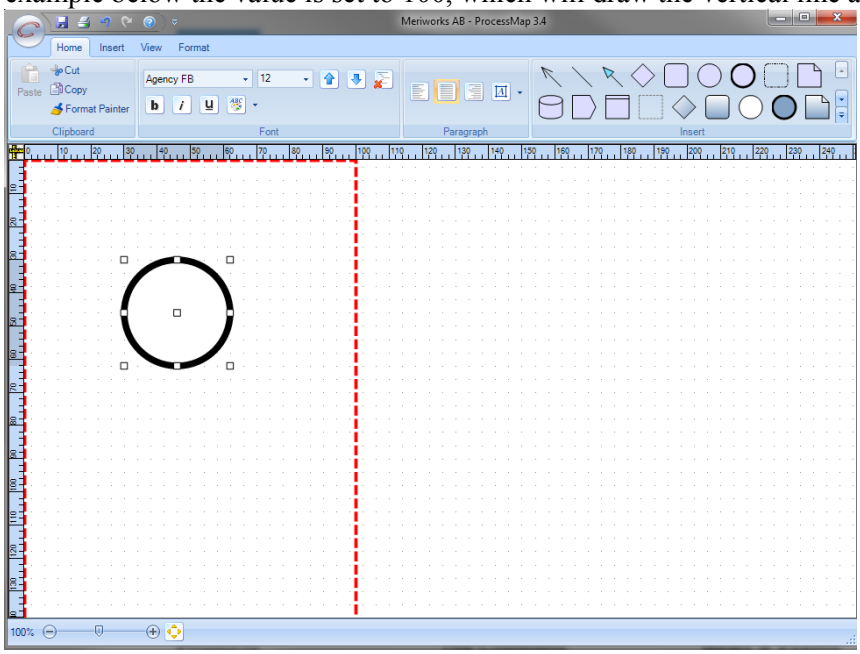
		Name	Field name	Type	Required	Localized	Searchable	Tab	From code
		MainBody	Main body	XHTML string (>255)		Yes	Yes	Content	Yes
		MetaTitle	Title	Long string (>255)		Yes	Yes	Metadata	Yes
		PageImage	Teaser image	Content Item				Content	Yes
		MetaKeywords	Keywords	String List		Yes		Metadata	Yes
		TeaserText	Teaser text	Long string (>255)		Yes	Yes	Content	Yes
		HideSiteHeader	Hide site header	Selected/not selected		Yes		Settings	Yes
		MetaDescription	Page description	Long string (>255)		Yes	Yes	Metadata	Yes
		HideSiteFooter	Hide site footer	Selected/not selected		Yes		Settings	Yes
		MainContentArea	Large content area	ContentArea			Yes	Content	Yes
		ProcessMap	ProcessMap	ProcessMap				Content	Yes
		DisableIndexing	Disable indexing	Selected/not selected		Yes		Metadata	Yes

You can then navigate to the ProcessMap property and go to the Custom Settings tab to set/edit the property settings.



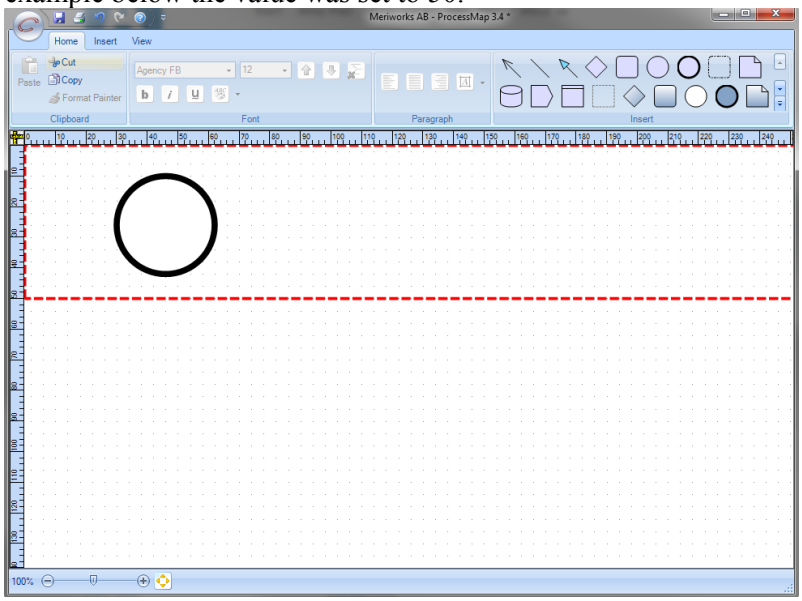
8.1. Vertical line

If this values is set a red, vertical line will be drawn in the ProcessMap editor when editing the map. In the example below the value is set to 100, which will draw the vertical line at 100 units.

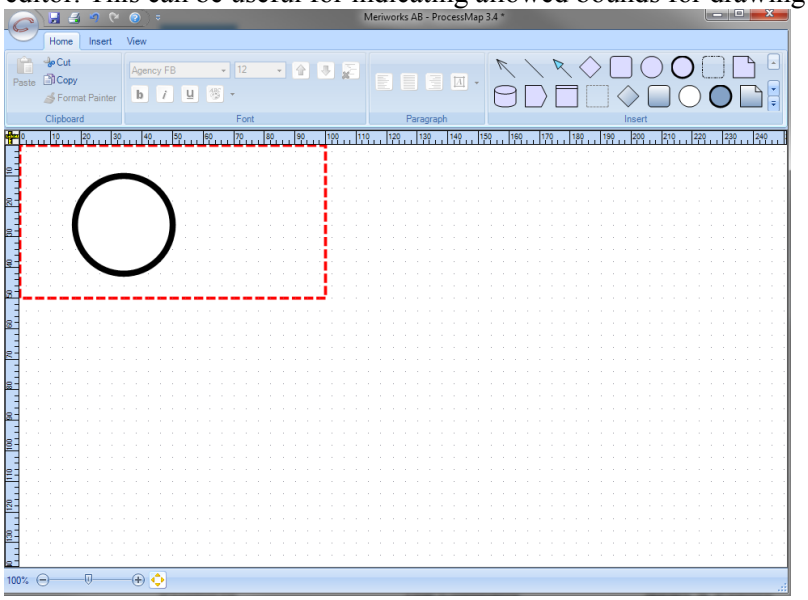


8.2. Horizontal line

This works just like the vertical line but will draw a horizontal line instead at the specified position. In the example below the value was set to 50.



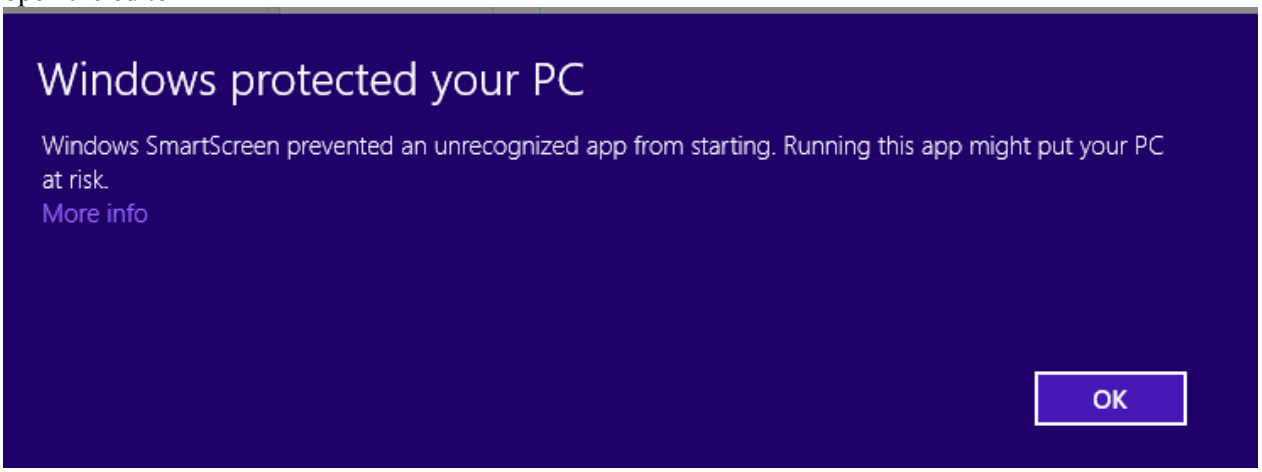
In the image below both vertical and horizontal lines are combined to display a rectangular area in the editor. This can be useful for indicating allowed bounds for drawing the process maps.



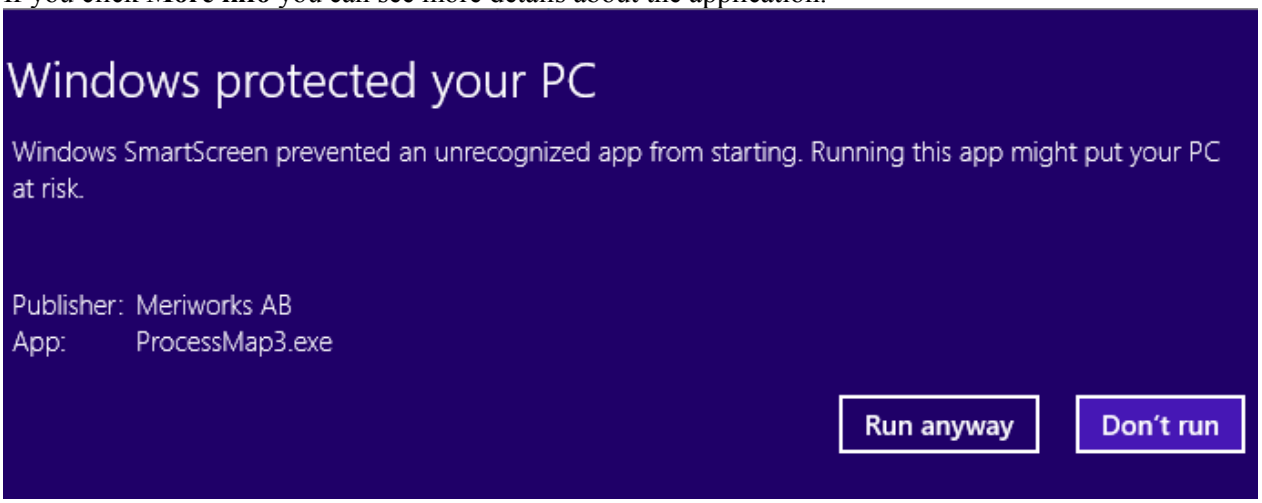
The horizontal and vertical lines will not be visible when rendering the resulting image – they are only visible as guide lines inside the editor.

9. Windows SmartScreen warning

When using Windows 8 and greater, you may be presented with a dialogue like this when trying to open the editor



If you click **More info** you can see more details about the application.



Verify that the application is signed by Meriworks AB and click **Run anyway** to start ProcessMap.

More information on Microsoft SmartScreen filter can be found on Microsofts web page.

10. Security configuration modifications

The final step of configuring the ProcessMap for usage includes the following steps:

10.1. Server settings

If you experience errors regarding security access please read chapter 11,"Authentication modes and security settings", for further assistance. This also applies if you are not using windows authentication.

10.2. Client settings

10.2.1. .Net framework

For the ProcessMapEditor to work properly each client has to have Microsoft .NET framework installed. The version of the framework depends of which version of ProcessMap the server is running (see 2.1).

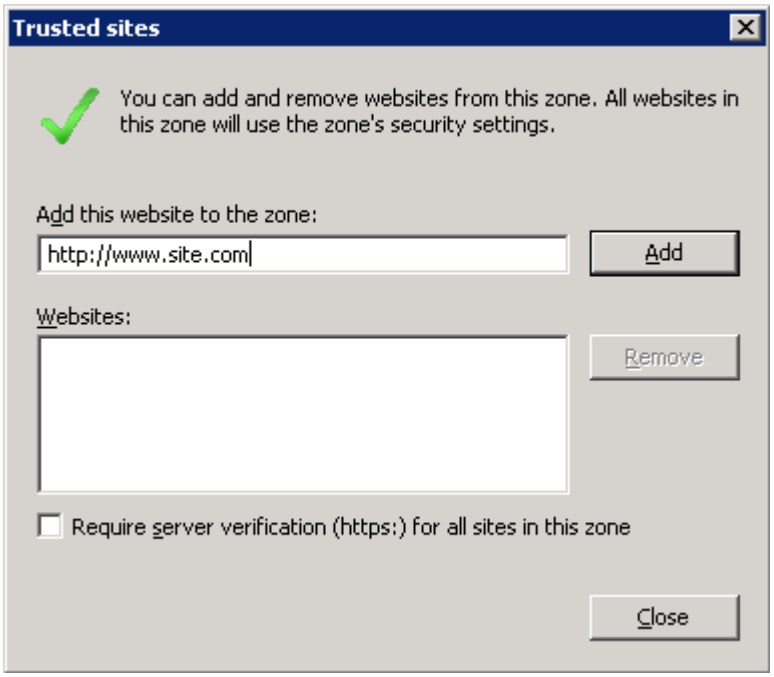
For more information about Microsoft .NET: http://www.microsoft.com/net

10.2.2. Internet Explorer Security Zone

If you use Windows Vista you also have to make sure that the EPiServer web server is added as a trusted site. Check the setting in the browser system tray.

Proceed in the following way to add the server url to the Trusted Sites:

Select the internet explorer menu "Tools->Internet Options-> Security->Trusted Sites->Sites".



Enter the server URL in the upper field, click add followed by Close.
The new setting can now be seen in the right bottom corner of Internet Explorer:

11. Authentication modes and security settings

Depending on the configuration of EPiServer, different Authentication methods are possible. The optimal configuration for ProcessMap is using either Windows authentication or forms authentication.

11.1. Security items

The following resources are vital to the ProcessMap system and it is important that the security regarding these resources is applied correctly.

11.1.1. ProcessMap Client files

The ClickOnce process that is responsible for downloading and launching ProcessMap makes anonymous requests to the Client folder. (/Plugins/ProcessMap/Client)

It is therefore important that anonymous access is **enabled** for these files. This is especially notable in environments like intranets, where it is common to apply global deny permissions for anonymous access.

In addition to this the anonymous user identity must also have read access to this folder on disk. (The anonymous user identity can be set through IIS, the default identity is IUSR)

11.1.2. ProcessMapService.asmx

This is the webservice used by the ProcessMap editor for getting language data, loading/saving ProcessMap data and getting EPiServer page information. Therefore it is recommended that the authenticated user is used for accessing the service.

11.2. Authentication modes

ProcessMap supports the following authentication modes:

11.2.1. Windows authentication

If windows authentication, is used accessing all resources should be done in the windows user context. E.g. the currently logged on windows user should match the EPiServer user. The ProcessMap Client may have problems with authentication if these differ, for example if you log in to a machine with a specific VPN account and then sign in to EPiServer with a different windows account to launch ProcessMap.

The following settings must be set:

The ProcessMapService.asmx must **deny** anonymous access.

The user should have read access to the EPiServer pages that should be visible selectable in the link editor.

11.2.2. Forms authentication

If forms authentication is used, accessing all resources should be done after starting a forms authenticated session. The client editor will use the authentication cookie to authenticate its web service calls.

The following settings must be set:

- The ProcessMapService.asmx must **deny** anonymous access (done in web.config in plugins/processmap).
- The user should have read access to the EPiServer pages that should be selectable in the link editor.

By default the following settings are added to the installation. This configuration denies anonymous access to both the editor and service.

12. Configuration

This section describes the configuration options that are available for ProcessMap. This section also describes all changes made to the web.config file upon installation.

12.1. EPiServer CMS 8

ProcessMap for EPiServer CMS 8 is installed as an EPiServer module, with the data stored in the App_Data folder and the ProcessMap application files and configurations are located in Plugins/ProcessMap. The configuration sections below is used for older versions of ProcessMap and EPiServer.

12.2. ProcessMapExtension section

For EPiServer CMS 7.5 and below, the ProcessMap has an own configuration section in the web.config and is located in:

```
configuration/se.meridium/processMapExtension
```

Below follows all configuration keys that exist in the section and a description of the usage and configuration possibilities.

12.3. ConfigSections

System settings, do not change.

```
<configSections>
   <sectionGroup name="se.meridium">
      <section name="ProcessMap"
type="ProcessMap.Server.Configuration.ProcessMapConfigurationSection,Proc
essMap.Server />
   </sectionGroup>
</configSections>
```

12.4. Modules and Handlers

12.4.1. EPiSever CMS version 4, 5 and 6

System settings for IIS 5 and IIS 6, do not change.

```
<system.web>
 <httpModules>
  <add name="ProcessMapExtension"
type="ProcessMap.EPiServerCMS.Extensions.ProcessMapExtension.Plugi
nStartup, ProcessMap.EPiServerCMS" />
 </httpModules>

 <httpHandlers>
   <add verb="GET,HEAD" path="*/ProcessMapImage.axd"
validate="false"
type="ProcessMap.Server.ProcessMapImageHandler,ProcessMap.Server"
/>
 </httpHandlers>
</system.web>
```

For IIS 7

```
<system.webServer>
   <modules>
      <add name="PluginStartup"
type="ProcessMap.EPiServerCMS.Extensions.ProcessMapExtension.Plugi
nStartup" />
   </modules>
   <handlers>
      <add name="*/ProcessMapImage.axd_GET,HEAD" verb="GET,HEAD"
path="*/ProcessMapImage.axd"
type="ProcessMap.Server.ProcessMapImageHandler,ProcessMap.Server"
/>
   </handlers>
   <validation validateIntegratedModeConfiguration="false" />
</system.webServer>
```

12.4.2. EPiSever CMS version 7

System settings for IIS 5 and IIS 6, do not change.

```
<system.web>
 <httpHandlers>
   <add verb="GET,HEAD" path="*/ProcessMapImage.axd"
validate="false"
type="ProcessMap.Server.ProcessMapImageHandler,ProcessMap.Server
/>
 </httpHandlers>
</system.web>
```

For IIS 7

```
<system.webServer>
   <handlers>
      <add name="*/ProcessMapImage.axd_GET,HEAD" verb="GET,HEAD"
path="ProcessMapImage.axd"
type="ProcessMap.Server.ProcessMapImageHandler,ProcessMap.Server
/>
   </handlers>
   <validation validateIntegratedModeConfiguration="false" />
</system.webServer>
```

12.4.3. EPiServer CMS version 8

In ProcessMap 3.4 the ProcessMapImage.axd handler is added only for paths starting with ~/Plugins/ProcessMap/ProcessMapImage.axd and this url will be used for all created urls. This is configured automatically in the plugins/processmap/web.config.

Note! If you upgraded from an old version of ProcessMap (prior to 3.4) you either need to keep the old handler configuration in web.config to allow the pre-rendered html to work or force the html to be rendered by removing the html files in the "~/Plugins/ProcessMap/ProcessMapData" folder.

12.5. Application Settings

Settings controlling the application. Should only be changed if the application or site is moved.

12.5.1. EPiServer CMS version 4, 5 or 6

```
<se.meridium>
    <ProcessMap>
      <appSettings>
        <add key="DataServiceFactory"
value="ProcessMap.EPiServerCMS.Core.DataServiceFactory,ProcessMap.EPiServ
er[*]/>
        <add key="processMapEditorBasePath"
value="http://url/Extensions/ProcessMapExtension/Client" />
        <add key="XMLDataPath"
value="C:\EPiServer\Sites\sitename\Extensions\ProcessMapExtension\Process
MapData\ProcessMap_XMLDefinitions" />
        <add key="TemplateFolder"
value="C:\EPiServer\Sites\sitename\Extensions\ProcessMapExtension\Process
MapItems" />
      </appSettings>
    </ProcessMap>
<se.meridium>
```

Note:
[*] The assembly name is version dependant:
CMS 4 – ProcessMap.EPiServer4
CMS 5 – ProcessMap.EPiServerCMS
CMS 6 – ProcessMap.EPiServer6

12.5.2. EPiServer CMS version 7

```
<se.meridium>
    <ProcessMap>
      <appSettings>
        <add key="DataServiceFactory"
value="ProcessMap.EPiServerCMS.Core.DataServiceFactory,ProcessMap.EPiServ
er[*] />
        <add key="processMapEditorBasePath"
value="http://url/Plugins/ProcessMap/Client" />
        <add key="XMLDataPath"
value="C:\EPiServer\Sites\sitename\Plugins\ProcessMap\ProcessMapData\Proc
essMap_XMLDefinitions" />
        <add key="TemplateFolder"
value="C:\EPiServer\Sites\sitename\Plugins\ProcessMap\ProcessMapItems" />
      </appSettings>
    </ProcessMap>
<se.meridium>
```

Note:
[*] The assembly name is version dependant:
CMS 7.1 – ProcessMap.EPiServer7
CMS 7.5 – ProcessMap.EPiServer75

12.5.3. EPiServer CMS version 8

From version 3.4, configuration is optional and as default values are shown below. Data is now stored below the App_data folder. If needed, you can add the configuration section to point to another location (like for instance if you have a load balanced setup where you need to share the data folders).

```
<se.meridium>
    <ProcessMap>
      <appSettings>
        <add key="XMLDataPath"
value="~/App_Data/ProcessMap/ProcessMap_XMLDefinitions" />
        <add key="TemplateFolder"
value="~/App_Data/ProcessMap/ProcessMapItems" />
      </appSettings>
    </ProcessMap>
<se.meridium>
```

12.6. ProcessMapBrokenLinkReporter

The link reporter will scan all content added to the symbols in the graphical ProcessMap editor for occurrences of broken links. A report file will be generated if any broken links are found.

Note! The current version of the link reporter does not validate process maps in blocks.

The following keys will need configuration:

```
<se.meridium>
    <ProcessMap>
      <appSettings>

<add key="ProcessMapPageTypeIDs" value="" />
<add key="BrokenLinksOutputDirectory" value="" />
<add key="BrokenLinksResponsibleMailAddress" value="" />
<add key="BrokenLinksFromMailAddress" value="" />
<add key="SendBrokenLinkReportsToEditor" value="false" />

      </appSettings>
    </ProcessMap>
<se.meridium>
```

Below follows an explanation of each key.

```
<add key="ProcessMapPageTypeIDs" value="" />
```

A list with all templates equipped with process maps. Add each template id as a comma separated string.

Note! It is possible to identify individual page type ids on the admin page. Select the "Page Type" tab and position the mouse pointer over the page type entry. A link will be displayed in the internet explorer status bar. The last digits are the page id. If for example the link http://www.yourorganisation.com/admin/EditPageType.aspx?id=20 is shown then will "20" be the pageid.

```
<add key=" BrokenLinksOutputDirectory " value="" />
```

Set the folder where the report will be saved.

Note! The path refers to the local web server file system. It is important that the root user of the website has write credentials to the report folder.

```
<add key="BrokenLinksResponsibleMailAddress" value="" />
```

Add an e-mail address of a person that will always get an email with the broken link report attached.

```
<add key="BrokenLinksFromMailAddress" value="" />
```

Add the senders e-mail address.

Note! The address must belong to a valid domain.

```
<add key="SendBrokenLinkReportsToEditor" value="false" />
```

Set if the specific page editor should receive the report as well (see BrokenLinksResponsibleMailAddress explained above). The mail will be sent to the most recent editor of the page.

Note! If neither the BrokenLinksResponsibleMailAddress nor the SendBrokenLinkReportsToEditor is set will the report only be saved to disk.

13. Create new symbols

It is possible to modify existing symbols or creating new ones. Symbols are stored as xml-files on the server in the folder "<webroot>/ProcessMapSymbols/". New symbols must have a unique id; avoid using id:s between 200 and 299. To make a new symbol appear in the Editor add the id to the "SelectedTemplates" node in the configuration file. See section 4 for more information about changing the configuration.

13.1. Symbol xml definition

13.1.1. "ProcessMapItemTemplate" node

This is the parent node that contains all the information that is needed to draw the symbol. It has the following attributes.

- **templateId** (integer) - A unique id for the symbol
- **zLayer** (integer) - The z-index of the symbol. Symbols with high z-index are placed in front of symbols with lower indexes.
- **resizable** (boolean) - Can the symbol be resized?
- **rotate** (boolean) - Can the symbol be rotated?
- **rotateContent** (boolean) - Should the text in the symbol rotate when the symbol is rotated?
- **linkDisabled** (boolean) - Should the possibility to add links to the symbol be disabled?
- **affectRouting** (boolean) – Should the symbol affect routing of lines, i e make lines be routed round it. If not it will be possible to draw lines across the symbol.
- **defaultRotation** (integer) - How many degrees the symbol should be rotated by default (clockwise).
- **minimumWidth** (integer) - Minimum width in pixels.
- **minimumHeight** (integer) - Minimum height in pixels.
- **imageAlign –** the behavior of a shapes image. Only needed if the image node is used in a shape. The value can be stretch, fit, center, tile, topLeft, topCenter, topRight, middleLeft, middleRight, bottomLeft, bottomCenter or bottomRight.

```
<ProcessMapItemTemplate templateId="206" zLayer="10"
resizable="true" rotate="false" rotateContent="true"
linksDisabled="false" affectRouting="true" defaultRotation="0">
<!-- Other nodes for the symbol's definition are placed here -->
</ProcessMapItemTemplate>
```

13.1.2. "NameDictionary" node

This node contains the symbols names for different languages. The key is a language code and the value is the name.

```
<NameDictionary>
  <item>
    <key><string>en</string></key>
    <value><string>Database</string></value>
  </item>
  <item>
    <key><string>sv</string></key>
    <value><string>Databas</string></value>
  </item>
</NameDictionary>
```

13.1.3. "ShapePlaceHolder" node

This node contains the shape of the symbol. The shape is split up in three child nodes. The "outline" node that specifies the outer shape of the symbol, the "decoration" node that specifies the lines within the symbol and the "textarea" node which defines the area where text can be written. The shapes are defined by using the following elements.

- **line -** a single line from one point to another.
- **round-rectangle** - draws a circle from a starting point, width, height and radius.
- **bezier** - draws a curved line through the specified coordinates.
- **arc** - draws a part of a circle from a starting point, a starting angel and a sweep angel.

This is an example of how to create a rectangle with a text area. The fill-mode attribute can be *Winding* or *Alternate* and determines how adjacent areas in the symbol are filled. It will not have any affect in this example since this symbol only has one area.

```
<ShapePlaceholder>
  <Shape id="" fill-mode="Winding">
    <outline>
      <line dash-style="Custom" width="-1">
        <point x="0" y="100" />
        <point x="100" y="100" />
      </line>
      <line dash-style="Custom" width="-1">
        <point x="100" y="100" />
        <point x="100" y="0" />
      </line>
      <line dash-style="Custom" width="-1">
        <point x="100" y="0" />
        <point x="0" y="0" />
      </line>
      <line dash-style="Custom" width="-1">
        <point x="0" y="0" />
        <point x="0" y="100" />
      </line>
    </outline>
    <textarea>
      <line dash-style="Custom" width="-1">
```

```
          <point x="1" y="1" />
          <point x="1" y="99" />
        </line>
        <line dash-style="Custom" width="-1">
          <point x="1" y="99" />
          <point x="99" y="99" />
        </line>
        <line dash-style="Custom" width="-1">
          <point x="99" y="99" />
          <point x="99" y="1" />
        </line>
        <line dash-style="Custom" width="-1">
          <point x="99" y="1" />
          <point x="1" y="1" />
        </line>
      </textarea>
    </Shape>
  </ShapePlaceholder>
```

It is also possible to add a background image to the symbol. The value of the node should be a base64 encoded image.

```
  <image x="0" y="0" width="100" height="100">
       iVBORw0KGgoAAAANSUh……
  </image>
```

13.1.4. "Connectors" node

This node contains connection points for the symbol. A connection point is the point where a line can be attached. The connection point is defined as a "ProcessMapItemConnector" node and has an x and y coordinate. ConnectorId and connectorType are not needed.

```
  <Connectors>
            <ProcessMapItemConnector connectorId="1" x="0" y="50"
connectorType="Up">
                  <Documents />
            </ProcessMapItemConnector>
  </Connectors>
```

13.1.5. "Brush" node

This node is used to set a background color on a node.

```
  <Brush d2p1:type="SolidBrushTemplate"
xmlns:d2p1="http://www.w3.org/2001/XMLSchema-instance">
            <ColorString>ARGB(255,255,255,255)</ColorString>
  </Brush>
```

13.1.6. "Shadow" node

This node specifies if the symbol should have a shadow. It is set to true as default.

```
<Shadow>false</Shadow>
```

13.1.7. "Aspect Ratio" node

This node is used for giving the symbol a certain aspect ratio.

```
<AspectRatio>0.65</AspectRatio>
```

14. Property Rules

It's possible to create rules that can affect the appearance and behavior of the processmap.
A rule consists of name, condition and effect. When the processmap is rendered the rules is applied to all symbols and if the conditions are met the effect is applied.

Examples:

```
<PropertyRule name="Links">
  <Condition>Symbol.Links > 0</Condition>
  <Effects>
    <ArrayOfPropertyEffect>
      <PropertyEffect xsi:type="ShadowPropertyEffect" enabled="true" />
    </ArrayOfPropertyEffect>
  </Effects>
</PropertyRule>
```

Any symbol with one or more links will be show with a shadow.

```
<PropertyRule name="HideLinks">
  <Condition>Symbol.Links.IsPropertySet( Hidden )</Condition>
  <Effects>
    <ArrayOfPropertyEffect>
      <PropertyEffect xsi:type="HideLinkEffect" enabled="true" />
    </ArrayOfPropertyEffect>
  </Effects>
</PropertyRule>
```

Any symbol with the property "**Hidden**" will not be shown in the popup menu when the symbol is clicked.

```
<PropertyRule name="Test">
  <Condition>symbol.links.isPropertySet(Important)</Condition>
  <Effects>
    <ArrayOfPropertyEffect>
      <PropertyEffect xsi:type="ColorPropertyEffect"
        location="SymbolBorder" color="RGB(0,0,0)" />
```

```
    </ArrayOfPropertyEffect>
  </Effects>
</PropertyRule>
```

Any symbol with the property "**Important**" will be shown with a black border regardless of what color the symbol normally have.

Location can be either SymbolBackground or SymbolBorder.

14.1. Conditions

The condition is specified using a query language that should return a true or false statement. The syntax is described below.

Symbol.ID == 1001
Is true if the symbol has id 1001

Symbol.Links > 5 AND Symbol.Links.IsPropertySet('My property')
Is true if the symbol has more than 4 links and any of the links has the property "My property".

```
BNF for PropertyCondition language
http://cui.unige.ch/db-research/Enseignement/analyseinfo/AboutBNF.html

<question> ::= <questionStatement> [ <questionStatementSeparator>
<questionStatement>]

<questionStatement> ::= <propertyCondition> | <propertyConditionNode>

<questionStatementSeparator> ::= <andSeparator>|<orSeparator>

<propertyCondition> ::= <propertyConditionMethod> |
<propertyConditionPredicate>

<propertyConditionNode> ::= ( <questionStatement>
<questionStatementSeparator> <questionStatement> )

<andSeparator> ::= AND | <ampersand> | <ampersand><ampersand>

<orSeparator> ::= OR | <vertical bar> | <vertical bar><vertical bar>

<propertyConditionMethod>::= Symbol.Links.IsPropertySet(<propertyName> |
<quotedPropertyName>)

<propertyConditionPredicate> ::= <propertyValue> <propertyPredicate>
<propertyValue>

<ampersand> ::= &

<vertical bar> ::= |

<propertyName> ::= string

<quotedPropertyName> ::= <quote><propertyName><quote>
```

```
<propertyValue> ::= <numericPropertyValue> | <propertyVariable>

<propertyPredicate> ::= <propertyPredicateEqual> |
<propertyPredicateNotEqual> | <propertyPredicateGreater> |
<propertyPredicateLess> | <propertyPredicateGreaterOrEqual> |
<propertyPredicateLessOrEqual>

<quote> ::= '

<numericPropertyValue> ::= numeric value

<propertyVariable> ::= <symbol Id variable> | <number of links on symbol>

<propertyPredicateEqual> ::= = | ==

<propertyPredicateNotEqual> ::= != | <>

<propertyPredicateGreater> ::= >

<propertyPredicateLess> ::= <

<propertyPredicateGreaterOrEqual> ::= >=

<propertyPredicateLessOrEqual> ::= <=

<symbol Id variable> ::= Symbol.ID

<number of links on symbol variable> ::= <number of links on symbol
variable alt1> | <number of links on symbol variable alt2>

<number of links on symbol variable alt1> ::= Symbol.Links

<number of links on symbol variable alt2> ::= Symbol.Links.Count
```