XWiki Enterprise 5.4.4 Developer Guide

Table of Contents

Programming Overview

Reference Guide

- XWiki Data Model
- XWiki Scripting
- XWiki API Reference
- XWiki Widget Reference
- XWiki Macros Reference
  - XWiki Rendering Macros
  - XWiki Velocity Macros
  - XWiki Rendering Macros in Java
- Create and Override a Skin
- Skin Extensions
- Configure Keyboard Shortcuts
- Customize the PDF and RTF Export Look&Feel
- The XWiki Database Schema
- XWiki Query Guide
- Solr Schema and API
- Groovy Notifications
- Post Processing Groovy Listener
- XWiki Platform Plugins

XWiki API Reference

XWiki APIs are the methods that the XWiki development team consider safe to use because backward compatibility is guaranteed. They can be called from Java or directly from your wiki pages using a scripting language.

The XWiki **Scripting Reference Documentation** is available at this link.

The latest scripting API documentation is available as JavaDoc:

- Javadoc for XWiki Platform Core - 5.4.4

You might also be interested in the Rendering Engine API:

- Javadoc for XWiki Platform Rendering API - 5.4.4

The full Javadoc is available at this link. In case you are interested in older Javadocs of XWiki, they can be found in our Maven Release repository.

It is not recommended to use a class having `internal` in its package because it means that the class is not a public API and the XWiki developers can change it at any time, thus its stability is not guaranteed.

When you see code marked with the `@Unstable` it means it is a new public API that is still considered unstable and that can change at any time.

Related Pages

- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki Velocity Macros
  - XWiki Scripting
  - XWiki Rendering Macros in Java
  - XWiki Rendering Macros
  - XWiki Query Guide
  - XWiki Platform Plugins
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - User Avatar Macro
  - Translation Macro
  - The Watchlist Plugin

XWiki Data Model

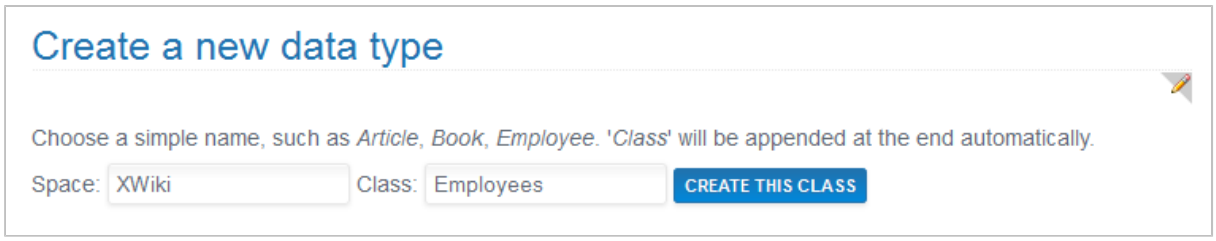
XWiki exposes a flexible data model that can be leveraged at the presentation level. Combined with its powerful presentation level scripting capabilities, XWiki Enterprise data model can be used for building simple to complex Web applications with little or no need to access the XWiki core. In other words, it is possible to build custom applications through the XWiki web interface, without having to compile, package and deploy software components.

XWiki Classes

An XWiki class defines a unique type of object, i.e. the properties an object can have. When defining a custom class, the application may need to create one or many objects, or instances of that class. A class is attached to a page and there can be at most one class per page. The class name is the name of the document containing that class.

Create a Class

The Class Editor Wizard has been developed for creating classes and is available by navigating to "XWiki.XWikiClasses". To create a class, enter its name and the space name in the "Create a new data type" section, then click on "Create this class".



You will be redirected to the class page in "Wiki" edit mode which will contain the following code:

```
{{velocity}}
## Replace Main with the Space where you want your documents to be created.
## Replace the default parent with the one of your choice and save the document.
##
#set($defaultParent = $doc.fullName)
#set($defaultSpace = 'Main')
{{/velocity}}
```

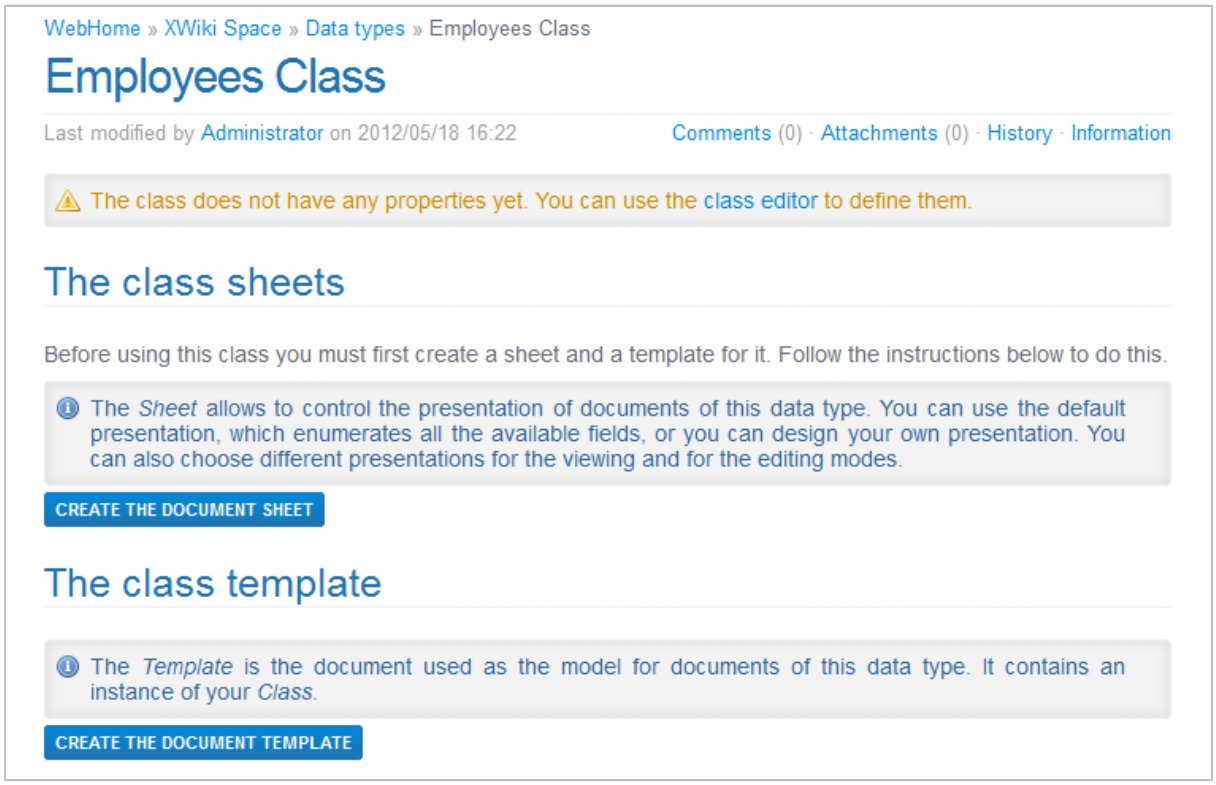
Change the default space which is "Main" with the name of the space where you want your pages to be created in, as the commented instructions in the page code suggest.

```
#set($defaultSpace = 'XWiki')
```

You can also change the default parent of the new documents by replacing the value of the `$defaultParent` parameter with the full name of your chosen document. Thus, the corresponding code should look like this:

```
#set($defaultParent = 'XWiki.WebHome')
```

Click on the "Save & View" button to finalize the creation of the "Employee" class.



Add Properties to the Class

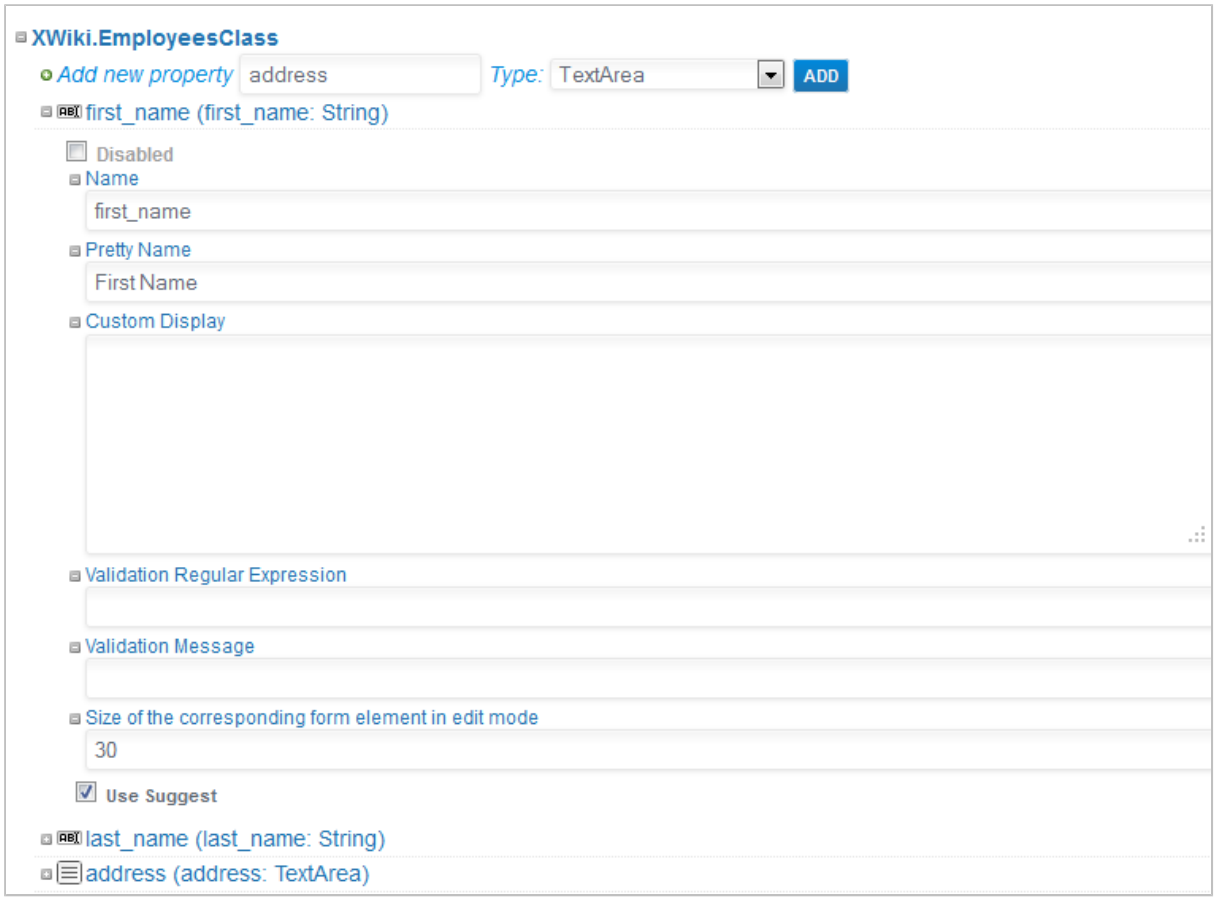
Below the page title, you should see the words "The class does not have any properties yet. You can use the class editor to define them".

Properties are characteristics that an object can have. In a class definition, the properties define the data fields that each unique instance of the class can have values for. XWiki currently supports the following kinds of properties (or datatypes):

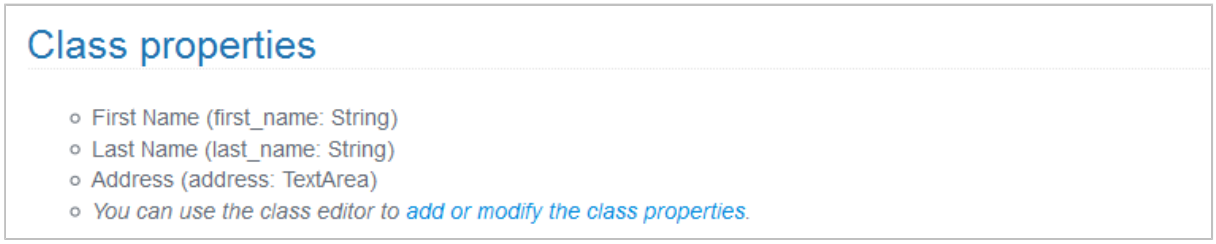
- Number
- String
- TextArea
- Password
- Boolean
- Email
- Static List
- Database List
- Database Tree
- Date
- List of Users
- List of Groups
- Page (the equivalent of "List of Documents")
- Access Right Levels
- Computed psuedo-field

Click on the "class editor" link to edit the "Employee" class in "Class" mode. Note that the link trail in the header is "Wiki Home » XWiki Space » Data types » EmployeeClass".

Now you need to add 2 "String" properties corresponding to the first name and last name of each employee and one "TextArea" property for the address. To do so, enter the name of the property in the "Add new property" text box, choose its type from the "Type" drop-down box then click on the "ADD" button.



Once you are done adding and configuring the properties, click on "Save & View". Your class page should look like this:



Objects

Objects are unique instances of a class with unique values defined for each of the properties that make up the class. An object is attached to a specific page and each page can have multiple objects. By using objects, structured information is inserted into the wiki, while in a traditional wiki it is only possible to enter unstructured information (plain text).

Sheets

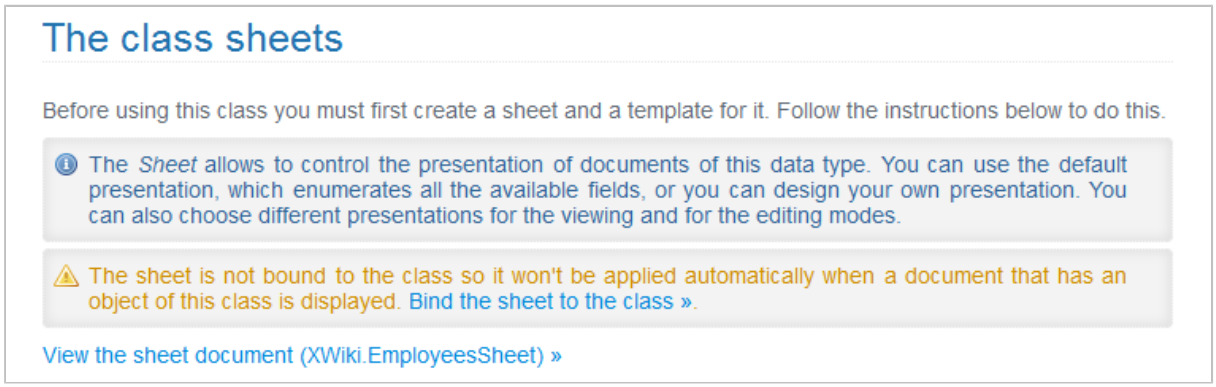
A sheet is an XWiki page determining how to display the objects, the title and the content of other XWiki documents. The title and the content are rendered in the context of the displayed document. Note that a sheet is not responsible for the page layout.

Create the Class Sheet

The class sheet displays the data stored in an object attached to an XWiki page. To control the display, you need

- to bind the XWiki class defining the object type to a sheet by adding a "XWiki.ClassSheetBinding" object to the class document
- to set the sheet reference as the value of the "Sheet" property.

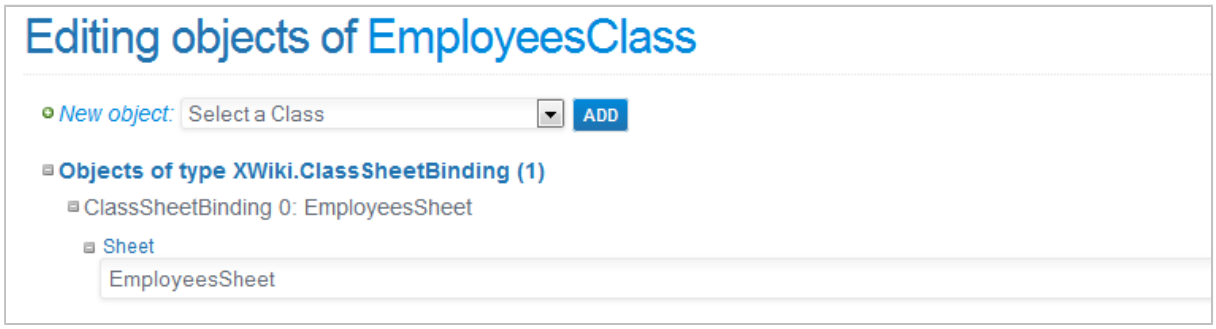
Click on "Create the document sheet" and notice that the content of the heading has changed.



Click on "Bind the sheet to the class" to add an "XWiki.ClassSheetBinding" object to the sheet page. If you then click on "View the sheet document (XWiki.EmployeesSheet)" and edit the page in "Wiki" mode, you will see the following code:

```
{{velocity}}
## You can modify this page to customize the presentation of your object.
## At first you should keep the default presentation and just save the document.
#set($class = $doc.getObject('XWiki.EmployeesClass').xWikiClass)
#foreach($prop in $class.properties)
  ; $prop.prettyName
  : $doc.display($prop.getName())
#end
{{/velocity}}
```

If you navigate back to the class page and edit it in "Objects" mode, you will see the attached "XWiki.ClassSheetBinding" object for which the value of the "Sheet" property is "EmployeesSheet". So "XWiki.EmployeesSheet" will be applied each time a wiki page contains an "XWiki.EmployeesClass" object.



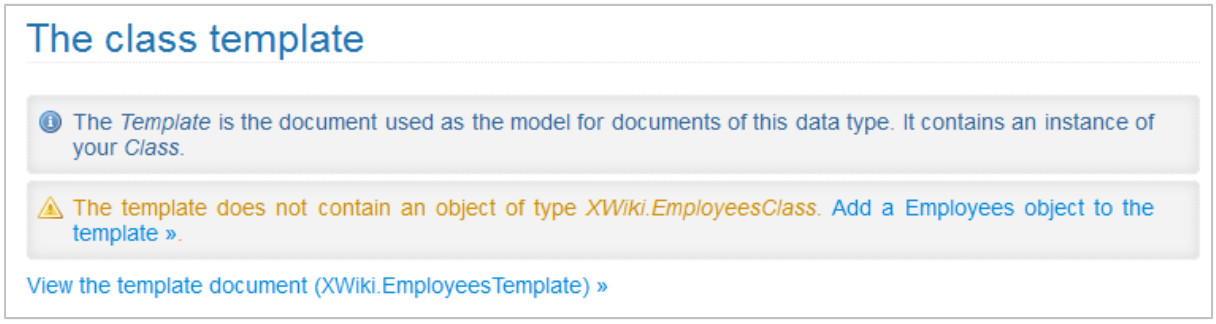
Document Sheets

Note that your custom class contains a second object, "XWiki.DocumentSheetBinding". A document sheet is useful when you wish a display a document differently than the rest of its type. In previous versions of XWiki you would have overridden the class sheet. Now you can bind your document to a custom sheet, by adding an "XWiki.DocumentSheetBinding" object and by setting the custom sheet reference as value of the "Sheet" property.

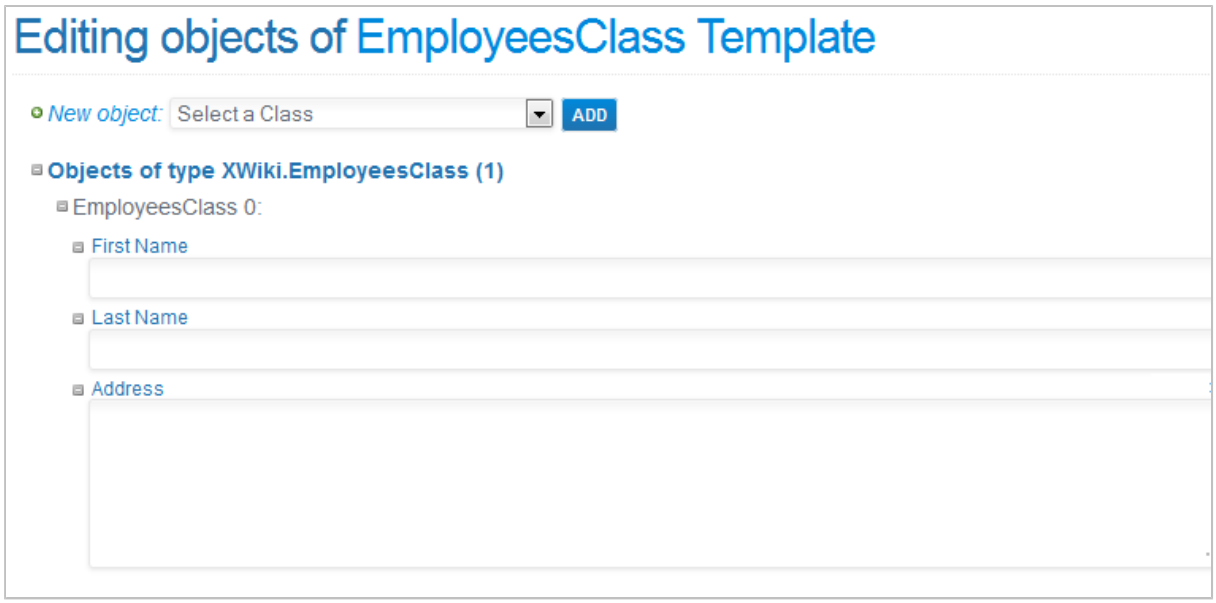
You may leave the "Sheet" property blank, which means that the document will be bind to itself, thus will control its own display.

Create the Class Template

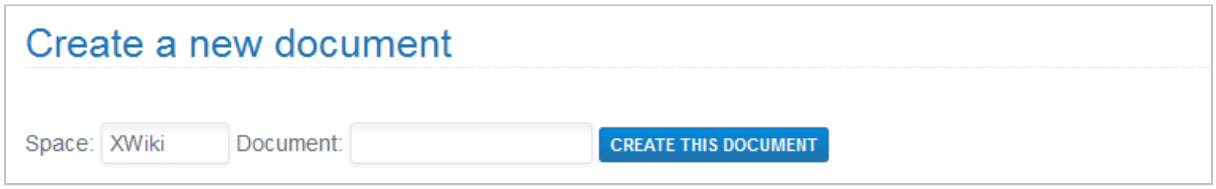
Navigate back to your class page and click on the "Create the document template" button. The Authoring Template will be automatically created but it doesn't contain an "XWiki.EmployeeClass" object yet. To add such an object, click on "Add a Employees object to the template".



If you then click on "View the template document (XWiki.EmployeesTemplate)" and edit the page in "Objects" mode, you will see the "XWiki.EmployeeClass" object attached:



To create employee forms, navigate back to the class page and search for the "Create a new document" section. Fill in the name of the form and the space you wish to create it into then click on "Create this Document".



The new form will open in "Inline" mode so that you can fill in the first name, last name and address. Click on "Save & View" to create your first Employee form.

Related Pages

- **User Guide**
  - Editing Modes
- **Programming Guide**
  - XWiki Scripting
  - Customize the Look and Feel of the PDF and RTF Export
  - Create and Override a Skin
- **Admin Guide**
  - The App Within Minutes Application
  - Customize the User Profile Sections

XWiki Scripting

Scrips allow you to create basic to complex web applications at the XWiki page layer without needing to compile code or deploy software components. In other words, the scripting syntax can be used in addition to wiki and HTML syntax as content of an XWiki page.

XWiki integrates jsr-223 scripting. There are several available languages by using one of the following macros:

- Velocity Macro - installed by default in XWiki Enterprise
- Groovy Macro - installed by default in XWiki Enterprise
- Python Macro - installed by default in XWiki Enterprise
- Ruby Macro - not installed by default in XWiki Enterprise
- PHP Macro - not installed by default in XWiki Enterprise

XWiki Scripting API

Bindings

Some bindings are automatically provided:

Name	Class	Description
xwiki	com.xpn.xwiki.api.XWiki	Represents the XWiki object.
xcontext	com.xpn.xwiki.api.Context	Represents the Context of the request.
request	com.xpn.xwiki.web.XWikiRequest	The servlet request. Generally used to get URL parameters.
response	com.xpn.xwiki.web.XWikiResponse	The servlet response.
doc	com.xpn.xwiki.api.Document	The current document.
cdoc	com.xpn.xwiki.api.Document	This variable represent the current default document (not the translated) from which to access the Comments, Objects and Attachments.
tdoc	`com.xpn.xwiki.api.Document`	It represents the translated document matching the requested language
util	com.xpn.xwiki.api.Util	Utility APIs available to scripting environments under the `util` variable.
msg	com.xpn.xwiki.web.XWikiMessageTool	Provides a internationalization service based on key/property values.
syntaxFactory	org.xwiki.rendering.syntax.SyntaxFactory	Allows the creation of a wiki syntax that the user can use to enter wiki content.
officeimporter		Puts a reference to Office Importer in newly created velocity contexts.

XWiki Component Access

You may directly access the XWiki components by using the following Groovy snippet:

```
{{groovy}}
def greeter = com.xpn.xwiki.web.Utils.getComponent(org.xwiki.component.HelloWorld.class);
println greeter.sayHello();
{{/groovy}}
```

XWiki Core Access

Sometimes, the XWiki API doesn't provide the needed methods for your application. You may access the XWiki core, but this presents an increased security risk and requires programming rights to run. Using the core should be avoided if possible.

```
{{groovy}}
def xc = xcontext.getContext();
def wiki = xc.getWiki();
def xdoc = doc.getDocument();
{{/groovy}}
```

The Groovy snippet creates 3 new objects:

- The underlying XWikiContext behind the Context object: **`xc`**
- The underlying XWiki object which backs the **`xwiki`** object: **`wiki`**
- The underlying XWikiDocument behind the current Document: **`xdoc`**

Many of the methods in **`wiki`** and **`xdoc`** require an instance of the **XWikiContext**. This is the underlying xcontext **`xc`** not the API context **`xcontext`**.

Query the XWiki Model

You can query the full XWiki Model using a script. Check the Query Guide for more information.

Velocity Specific Information

Velocity is the only scripting language which can be used without Admin or Programming Rights. This means Velocity scripts can be saved using a *username* with less permission so that an exploit of the script is less of a security breach.
You may access the XWiki core from Velocity but this will require programming rights. Strictly speaking, protected APIs are only available when the page containing them was last saved by someone with programming rights.

In Velocity, classes cannot be imported and direct access to XWiki components cannot be gained, as shown here. This leaves you with the provided bindings (**NOTE**: In Velocity, these bindings all start with **`$`** as with all other Velocity variables).

For more information about programming in Velocity, refer to the Velocity User Guide.

The following tools are also available in addition to the bindings.

- List Tool: **`$listtool`**
- Number Tool: **`$numbertool`**
- Comparison Date Tool: **`$datetool`**
- Math Tool: **`$mathtool`**
- Escape Tool: **`$escapetool`**
- Sort Tool: **`$sorttool`**
- StringUtils: **`$stringtool`**
- Collections Tool: **`$collectionstool`**
- JSON Tool: **`$jsontool`**
- Regex Tool: **`$regextool`**

If you wish to add new Velocity tools you will need to edit your `xwiki.properties` file and follow the instructions in there.

To include Velocity scripts in other Velocity scripts, see the dedicated page.

Controlling Which Sections to Display

You can control whether to display or not the "Comments", "History", "Attachment" and "Information" sections by setting some Velocity variables to "no":

```
#set ($showcomments = "no")
#set ($showattachments = "no")
#set ($showhistory = "no")
#set ($showinformation = "no")
```

To remove them all you can set:

```
#set($docextras = [])
```

Information About the Current User

The following variables are shortcuts for checking various information for **the current user**:

- `$isGuest`: checks if the current user is `XWiki.XWikiGuest`
- `$isSuperAdmin`: checks if the current user is the special user `superadmin`
- `$hasComment`: checks comment rights on the current document
- `$hasEdit`: checks edit rights on the current document
- `$hasWatch`: checks whether the user is authenticated and the watch service is available
- `$hasAdmin`: checks admin rights on the current document
- `$hasSpaceAdmin`: checks the admin rights on the `XWikiPreferences` document of the current space
- `$hasGlobalAdmin`: checks admin rights on `XWiki.XWikiPreferences`
- `$hasCreateSpace`: checks edit rights on that page that does not exist, in a space that doesn't exist
- `$hasCreatePage`: checks edit rights on that page that does not exist, in the current space
- `$hasProgramming`: checks if the current user has programming rights
- `$isAdvancedUser`: advanced users: `superadmin`, users with the `usertype` property set to "Advanced", guest users with admin rights

Example:

```
{{velocity}}
#if ($hasAdmin)
  ## This link will only be visible to users that have admin rights on this document
  [[Do some admin action>>Some.Document]]
#end
{{/velocity}}
```

Information About the Current Wiki

The following variables are shortcuts for checking various information about the current wiki:

- `$isReadOnly`
- `$isInServletMode`
- `$isInPortletMode`

Groovy Specific Information

The following example demonstrates how to use a Groovy script to interact with Velocity code in the same page. This example performs a DNS lookup from the velocity variable `$hostname` and stores the result in the variable `$address`.

Objects can be passed back and forth between scripting languages by storing them in commonly available objects. One such object which only lasts the length of the request is the context object, known as **xcontext**.

```
{{velocity}}
#set($hostname = "www.xwiki.org")
Host Name: $hostname
$xcontext.put("hostname", $hostname)
{{/velocity}}
{{groovy}}
import java.net.InetAddress;
host = xcontext.get("hostname");
InetAddress addr = InetAddress.getByName(host);
String address = addr.getHostAddress();
xcontext.put("address", address);
{{/groovy}}
{{velocity}}
IP Address: $xcontext.get("address")
{{/velocity}}
```

Related Pages

- **Programming Guide**
  - XWiki Velocity Macros

- XWiki Query Guide
- XWiki Data Model
- XWiki API Reference
- Velocity Macro
- Programming Overview

XWiki Rendering Macros

Definition

The XWiki rendering macros allow you to develop reusable and distributable macro modules, without the need for compiling or packaging. A wiki macro is defined by creating a new wiki page and attaching an **XWiki.WikiMacroClass** object to it which contains the following properties:

- **Macro id**: The identifier used when invoking the macro.
- **Macro name**: The macro name that will be displayed by the Wysiwyg editor.
- **Macro description**: A short description of the macro that will be displayed by the Wysiwyg editor.
- **Default category**: Default category under which this macro should be listed.
- **Supports inline mode**: Whether the macro can be used in an inline context or not.
- **Macro Visibility**
- **Macro content type**: Whether this macro should support a body or not.
- **Content description**: A short description about the macro content that will be displayed by the Wysiwyg editor.
- **Macro code**: The actual wiki code that will be evaluated when the macro is executed. You can use any XWiki content as long as it has the same syntax as the document.

Visibility

A wiki macro can have one of the following visibility types:

- `Global` - in a multi-wiki environment, the macro will be available on all the wikis of the farm
- `Current Wiki` - the macro will only be available on the wiki it was created on
- `Current User` - the macro will only be available for its author

Required Rights

The required rights depend on the macro visibility:

- for a macro available globally, the macro author needs to have programming rights
- if the macro is only available on the current wiki, the macro author must have admin rights
- a macro available for the current user does not require special permissions, except for the edit rights

Parameters

In order for a wiki macro to have parameters, you need to add an **XWiki.WikiMacroParameterClass** object per parameter. The class contains several fields that allow you to define your parameter:

- **Parameter name**: Name of the parameter, referred when invoking the macro with parameters.
- **Parameter description**: A short description of the parameter, which will be made available on the Wysiwyg editor.
- **Parameter mandatory**: Indicates if this particular parameter is mandatory. The wiki macro will fail to execute if a mandatory parameter is missing.
- **Parameter default value**

Access a Macro Parameter

A macro parameter can be accessed from any scripting language within the macro code using the syntax: `$xcontext.macro.params.parameterName` where "parameterName" is the name of the parameter as explained in the above section.

Invocation

To test your wiki macro, use the following syntax:

```
{{macroId parameter1="<valueParameter1>" parameter2="<valueParameter2>" /}}
```

or insert it using the "Wysiwyg" editor.

Wiki Macros Bundled with XWiki Enterprise

- Activity Macro - Provides information about recent changes done in an XWiki instance or farm.
- Attachment Selector Macro - A picker associated with XWiki object properties of type String which offers a dedicated UI for selecting from the available attached files.
- Documents Macro - Displays a list of documents in a livetable.
- Space Index Macro - Lists the documents in a space.
- Spaces Macro - Creates an ordered list of all the spaces on the wiki.
- Tag Cloud Macro - Creates a Tag Cloud with tags from all the pages within the wiki.

Related Pages

- **Programming Guide**
  - XWiki API Reference
  - Tag Cloud Macro
  - Spaces Macro
  - Space Index Macro
  - Programming Overview
  - Documents Macro
  - Attachment Selector Macro
  - Activity Macro

Activity Macro

The {{activity /}} macro used by the Activity Stream UI is bundled with XWiki Enterprise and it allows you to keep better track of what is happening in the wiki or the farm. The displayed notifications include:

- page creation
- page edit
- page delete
- comments and annotations events
- attachments events
- the posts shared with other users via the {{shareMessage /}} macro
- wiki join/leave events
- wiki invitation events

Usage

You can either edit the page in "Wiki" mode and add the line below

```
{{activity entries="<entries>" subentries="<subentries>" wikis="<wikis>" spaces="<spaces>" authors="<authors>" tags="<tags>" minor="<true|false>" rss="<true|false>"/}}
```

or to insert the macro using the Wysiwyg editor.

Parameters

Name	Optional	Allowed values	Default value	Description
entries	YES	A numerical value	20	The number of entries to display the activity for.
subentries	YES	A numerical value	10	The number of events to display for each entry.
wikis	YES	A list of strings	If no value is specified, the macro displays events for the current wiki	A comma separated list of wiki domain names to display events for.
spaces	YES	A list of strings	If no value is specified, the macro displays events for all spaces	A comma separated list of spaces to display events for.
authors	YES	A list of strings each representing the full name (of full prefixed name) of a user page	If no value is specified, the macro displays modifications for all users	A comma separated list of wiki users whose modifications to display.
tags	YES	A list of strings	None	A comma separated list of tags to display events for. Only documents containing these specific tags will be handled.
minor	YES	A string set to "true" or "false"	false	A flag determining whether to display events that create minor versions or not.
rss	YES	A string set to "true" or "false"	false	A flag deciding whether to show the activity RSS link or not.

Examples

1. Display the activity for spaces "Main", "XWiki" and "Blog" and for the "Marketing" and "Projects" sub-wikis:

```
{{activity spaces="Main,XWiki,Blog" wikis="marketing,projects" /}}
```

2. Display the activity for the global user "JaneDoe":

```
{{activity authors="xwiki:XWiki.JaneDoe" /}}
```

3. Display the activity, including minor edits, for pages containing the tags "Company" and "Events":

```
{{activity minor="true" tags="Company,Events" /}}
```

Related Pages

- **User Guide**
  - View the List of Joined Sub-Wikis
  - User Status
  - User Profile
  - User Preferences
  - Activity Stream
- **Programming Guide**
  - XWiki Rendering Macros
  - XWiki API Reference
  - Tag Cloud Macro
  - Spaces Macro
  - Space Index Macro
  - Programming Overview
  - Documents Macro
  - Attachment Selector Macro
  - Activity Stream Plugin

Attachment Selector Macro

The "Attachment Selector" is a bundled picker associated with XWiki object properties of type String. The macro is used by default when choosing the user avatar and it offers a dedicated UI for selecting from the available attached files. After selecting an attachment, the property value becomes the name of that attachment.

The selector displays an attachment gallery which lists the attached files of the current document provided their format is an accepted one. Users can also upload new attachments, select or delete the existing attachments.

Usage

```
{{attachmentSelector classname="<classname>" property="<property>" cssClass="<cssClass>" savemode="<form|direct>"
buttontext="<buttontext>" defaultValue="<defaultValue>" displayImage="<true|false>" width="<width>" height="<height>"
alternateText="<alternateText>" link="<true|false>" targetdocname="<targetdocname>" /}}
```

Parameters

Name	Optional	Description	Accepted Values
classname	No	The full name of the document containing the XWiki class with the property associated with the picker.	-
property	No	The name of the class property associated with the picker.	-
object	Yes	The identifier (i.e. a number) of the object for which the property is displayed by this picker.	If missing, the first instance of the class (0) given by the parameter `classname` found in the document will be considered.
cssClass	Yes	A CSS class for the element surrounding the displayed property value.	-
savemode	Yes	Defines the way the property is updated	• `form` (the default value) meaning that the selected value is stored in an input that will be saved via an external form. • `direct` meaning that the update of the property value is done by the picker.
buttontext	Yes	The text of the button that triggers the picker	The default value is `$services.localization.render('xe.attachmentSelector.selectFile')`.
defaultValue	Yes	The name of the attachment displayed in view mode when the property is empty.	This parameter should be either empty or a reference to a wiki attachment.
filter	Yes	A comma separated list of file extensions accepted by the property.	If empty, all file formats are accepted.
displayImage	Yes	A flag stating whether images are displayed or whether only their name is printed.	• `true` • `false` (the default value)
width	Yes	The width of the displayed image which is only taken into account if `displayImage` is set to "true".	-
height	Yes	The height of the displayed image, only taken into account if `displayImage` is set to "true".	-

alternateText	Yes	The alternate text (corresponding to the `alt` attribute) of the displayed image, only taken into account if `displayImage` is set to "true".	-
link	Yes	A flag stating whether a link to the attachment is associated with the name/image of the attachment in view mode.	• `true` • `false` (the default value)
targetdocname	Yes	An optional reference to a document to use as the source and target for the attachments to display.	The name of the current document is used by default.

Examples

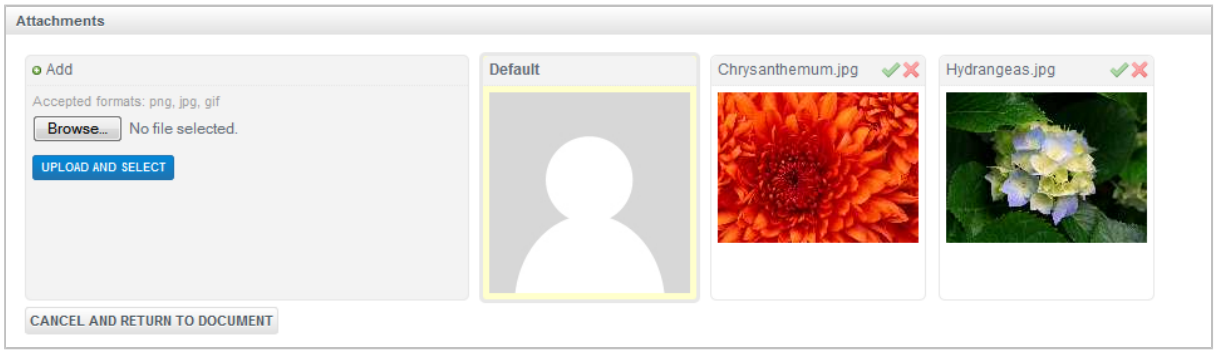
Example 1

{{attachmentSelector classname="XWiki.XWikiUsers" property="avatar" savemode="direct" displayImage="true" width="120" link="true" filter="png,jpg,gif" defaultValue="XWiki.XWikiUserSheet@noavatar.png"/}}

will display

Choose an attachment

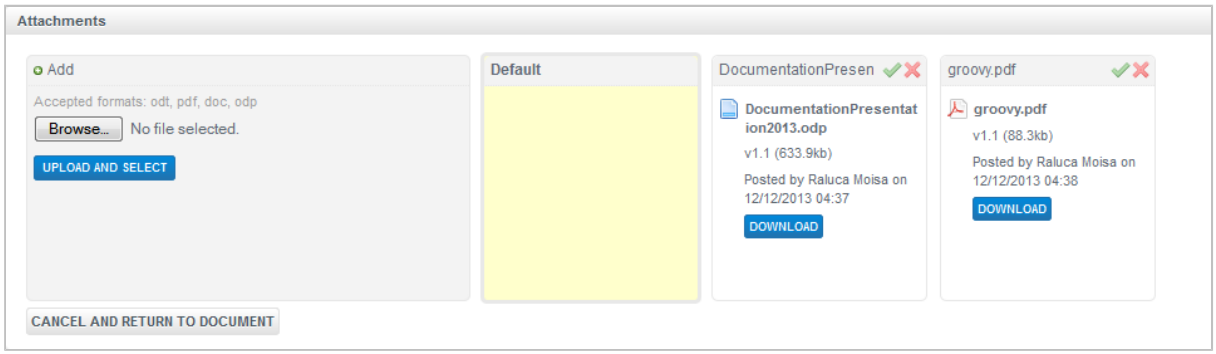
The XWiki class is "XWiki.XWikiUsers" and the allowed formats are "PNG", "JPG" and "GIF". The class property associated with the picker is "avatar" and the update of the property value is done by the picker. Also, the image width is "120 px" and in case no attached image is found, the "noavatar.png" image attached to "XWiki.XWikiUserSheet" will be displayed.



Example 2

{{attachmentSelector classname="XWiki.EmployeesClass" property="curriculumvitae" savemode="direct" displayImage="true" width="120" filter="odt,pdf,doc,odp" /}}

will display



Related Pages

- **User Guide**
  - Page Attachments
- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki Rendering Macros
  - XWiki API Reference
  - The "xwikidoc" Table
  - The "xwikiattrecyclebin" Table
  - The "xwikiattachment_content" Table
  - The "xwikiattachment_archive" Table
  - The "xwikiattachment" Table
  - Tag Cloud Macro
  - Spaces Macro
  - Space Index Macro
  - Programming Overview
  - Documents Macro
  - Activity Macro
- **Admin Guide**
  - WebDAV
  - Attach Large Files

Documents Macro

The Documents macro displays a list of documents in a livetable.

Usage

```
{{documents id="<id>" count="<count>" actions="<true|false>" space="<space>" parent="<parent>" columns="<columns>"
/}}
```

Parameters

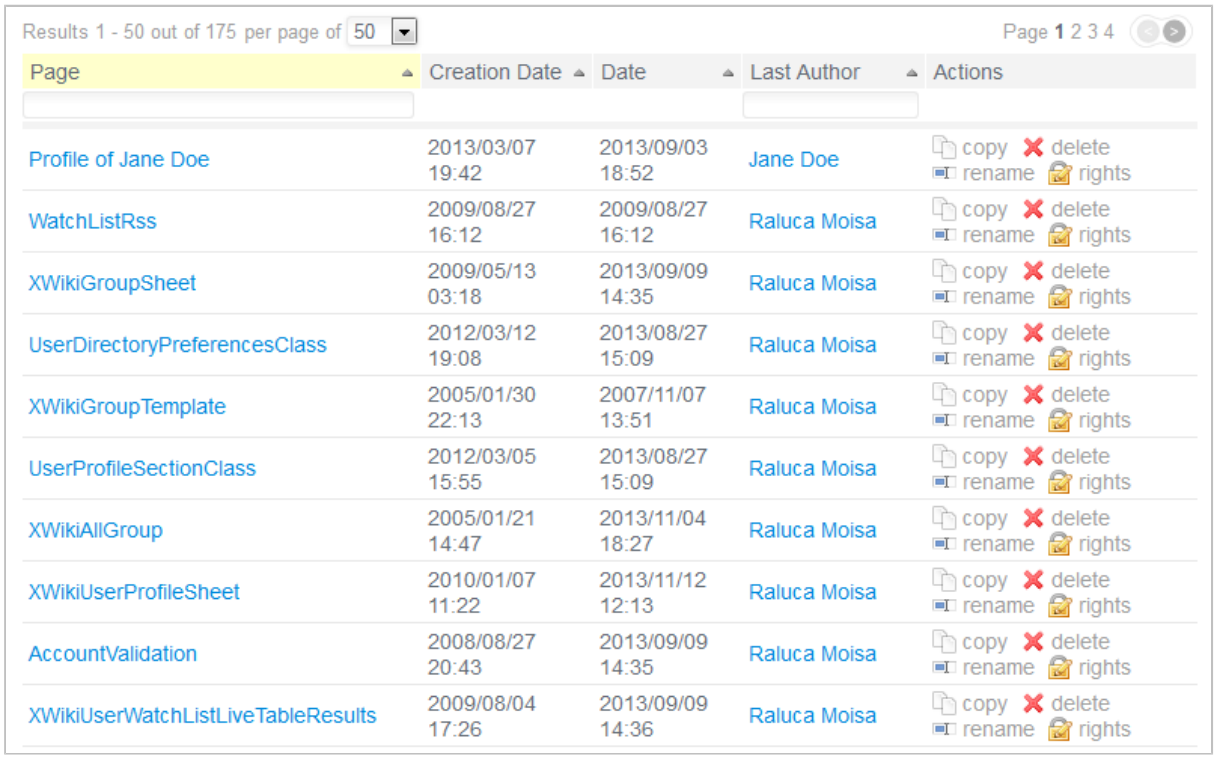
Name	Optional	Description	Accepted Values	Default Value
id	Yes	The HTML identifier of the livetable	A string	If no value is specified, a random value will be used
count	Yes	The number of items per page displayed by default	A number	15
actions	Yes	A flag stating whether to display the "Actions" column or not, depending on the user access rights	`true` or `false`	true
space	Yes	The name of the space to display documents for	A string	If no value is specified, the macro will display all documents
parent	Yes	The parent page to display documents for	A string	If no value is specified, the macro will display all documents
columns	Yes	The livetable columns to display	The name of a livetable column	"doc.name,doc.space,doc.date,doc.author"

Examples

Example 1

Display the documents in the "XWiki" space, 50 documents per page. The columns should be: the document title, the creation date, the last edit date, the last author and the "Actions" column.

```
{{documents id="MyCustomLivetable" count="50" actions="true" space="XWiki"
columns="doc.title,doc.creationDate,doc.date,doc.author" /}}
```



Example 2

Display all children of the "Main.WebHome" page.

```
{{documents id="DisplayChildren" parent="Main.WebHome" /}}
```

Related Pages

- **User Guide**
  - XWiki Dashboard Application
  - Index Application
- **Programming Guide**
  - XWiki Rendering Macros
  - XWiki Query Guide
  - XWiki API Reference
  - Tag Cloud Macro
  - Spaces Macro
  - Space Index Macro
  - Programming Overview
  - Livetable Macro
  - Attachment Selector Macro
  - Activity Macro

Space Index Macro

The "Space Index" macro is bundled with XWiki Enterprise and it is used for instance when creating a space with the "Space Dashboard" template. The macro displays the list of pages for a space passed as parameter, provided the user has view rights on the space.

The "Space Index" macro also displays pages marked as hidden.

Usage

```
{{spaceindex space="<space>" count="<count>" /}}
```

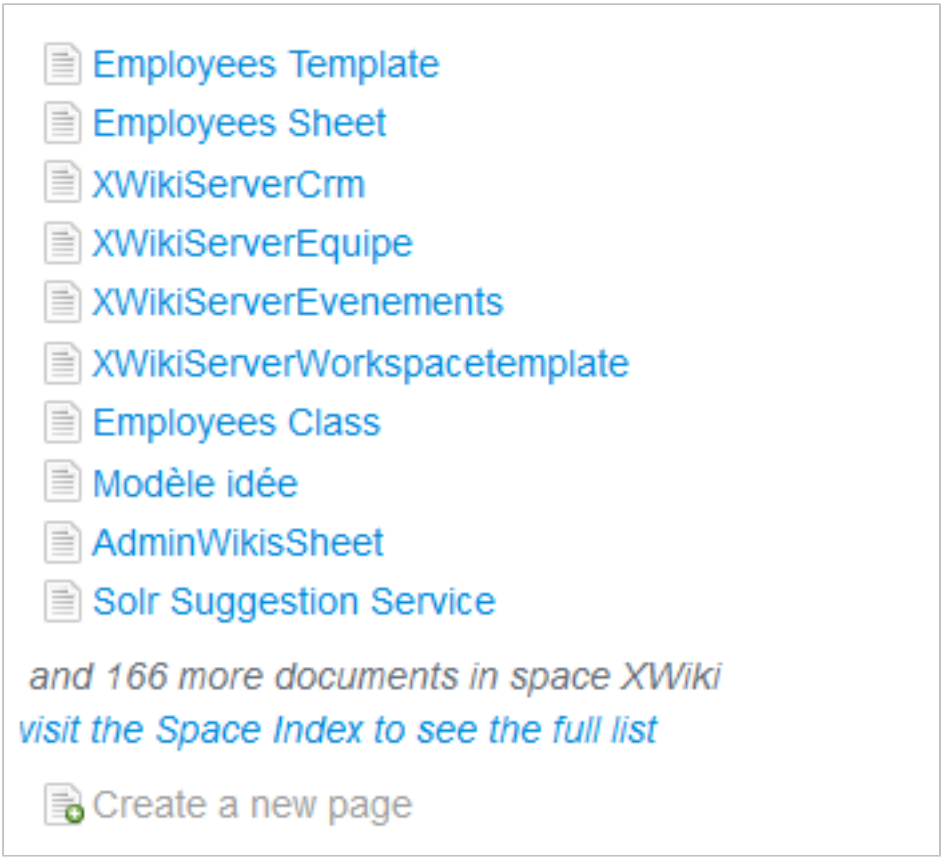
Parameters

Name	Optional	Allowed Values	Description	Default Value
space	Yes	Any string	The name of the space for which to display the document index.	If no value is specified, the macro will use the current space.
count	Yes	A number	The maximum number of documents to display. In case you want to display all documents, the value should be "0".	100

Example

```
{{spaceindex space="XWiki" count="50" /}}
```

will display



In case the space has more documents than the value of the `count` parameter, the macro will display a message similar to

and X more documents in space Y.

along with a link to the space index page.

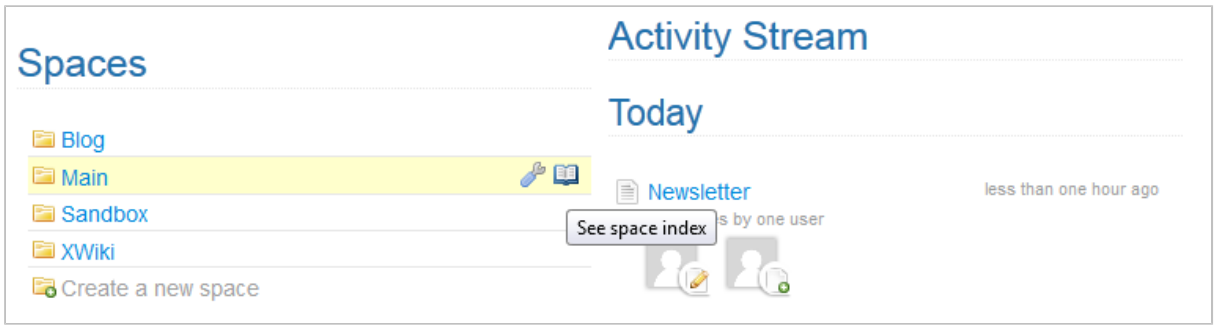
The "Create a new page" link will only appear for users having the necessary permissions like edit or admin.

Related Pages

- **User Guide**
  - Index Application
- **Programming Guide**
  - XWiki Rendering Macros
  - XWiki API Reference
  - Tag Cloud Macro
  - Spaces Macro
  - Programming Overview
  - Documents Macro
  - Attachment Selector Macro
  - Activity Macro

Spaces Macro

The "Spaces" macro is used by default on the wiki dashboard to display an ordered list of spaces, provided they are not marked as hidden.



Usage

{{spaces /}}

Related Pages

- **User Guide**
  - Index Application
- **Programming Guide**
  - XWiki Rendering Macros
  - XWiki API Reference
  - Tag Cloud Macro
  - Space Index Macro
  - Programming Overview
  - Documents Macro
  - Attachment Selector Macro
  - Activity Macro

Tag Cloud Macro

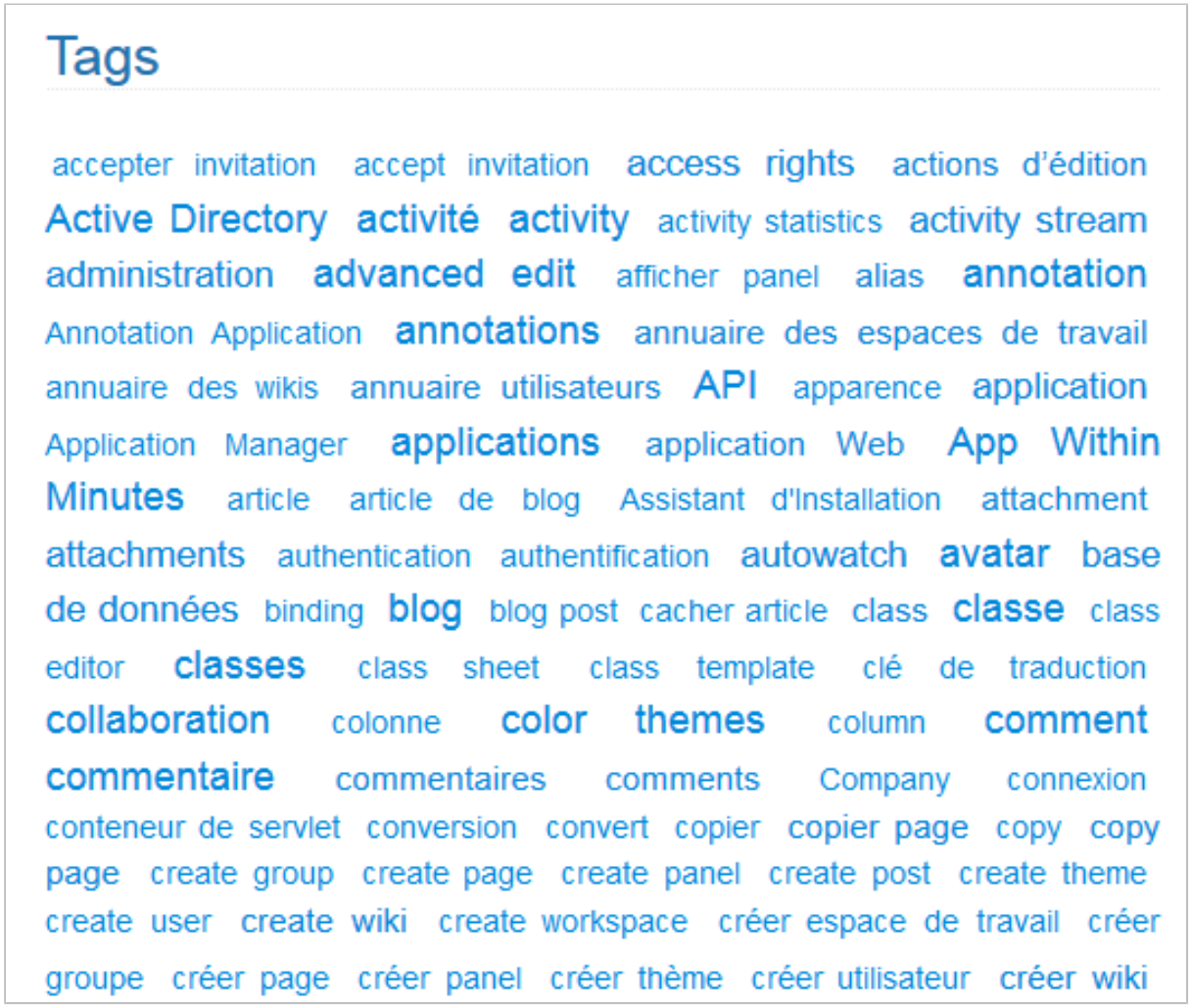
The Tag Cloud macro displays a Tag Cloud with tags from all the pages within the wiki.

Parameters

Name	Optional	Allowed values	Description	Default value
space	Yes	A string	The space to display the tag cloud for.	If missing, the tags in the whole wiki will be displayed.

Usage

{{tagcloud space="<space>" /}}



Related Pages

- **User Guide**
  - Tags Application
- **Programming Guide**
  - XWiki Rendering Macros
  - XWiki API Reference
  - Tag Plugin

XWiki Velocity Macros

XWiki Velocity macros are using the Velocity Template Language.

Description

Macro Definition

```
#macro(macroID $parameter1 $parameter2 ...)
//macro code
#end
```

Invocation

```
#macroID($parameter1 $parameter2 ...)
```

Velocity Macros Bundled with XWiki Enterprise

The bundled Velocity macros are defined in the "xwiki/templates/macros.vm" file.

- AIM Macro - Displays an online status indicator for an AIM account id. This macro is now marked as deprecated.
- **Display Panel Layout Macro** - Displays a list of panels in the provided number of columns. We are currently using the newer Container macro.
- **Error Macro** - Displays an error message in a styled box format, with an error icon. This macro is now marked as deprecated and replaced with the newer Error Macro.
- Flickr Macro - Displays a list of images taken from a Flickr RSS feed. This macro is now marked as deprecated.
- Flick User Macro - Displays the latest photos from a Flickr user. This macro is now marked as deprecated.
- Flickr User And Tag Macro - Displays latest public photos from a given user tagged with the given tag. This macro is now marked as deprecated.
- Flickr Tag Macro - Displays public photos from Flickr with the given tag. This macro is now marked as deprecated.
- Floating Box Macro - Displays a message in a styled floating box format.
- **Footnote Macro** - Adds a note of text placed (by default) at the bottom of the page. Currently, we are using the "xwiki/2.0" syntax Footnote Macro.
- IM Macro - Displays a XWiki User online status indicator depending on the "imaccount" and "imtype" properties in the user profile. This macro is now marked as deprecated.
- **Include Form Macro** - Includes a document which will be rendered in the context of the current document. This macro is now marked as deprecated and replaced with the newer Include macro.
- **Include In Context Macro** - Include a document which will be rendered in the context of the current document. This macro is now marked as deprecated and replaced with the newer Include macro.
- **Include Macros Macro** - Represents an alias for the "Include Form" macro. This macro is now marked as deprecated and replaced with the newer Include macro.
- **Include Topic Macro** - Includes text from another document. This macro is now marked as deprecated and replaced with the newer Include macro.
- **Info Macro** - Displays an informational message in a styled box format, using an information icon. This macro is now marked as deprecated and replaced with the newer Info macro.
- Jabber Macro - Displays an online status indicator for a given Jabber account id. This macro is now marked as deprecated.
- Livetable Macro - Displays a dynamic, filterable and sortable table of data.
- MIME Type Image Macro - Displays an icon for a variety of common mime types and file name extensions.
- MSN Macro - Displays an online status indicator for the given MSN messenger account id. This macro is now marked as deprecated.
- **Put Footnotes Macro** - Displays the footnotes mentioned so far in the wiki code and resets the footnote counter. Currently, we are using the "xwiki/2.0" syntax Put Footnotes Macro.

- Skype Macro - Displays an online status indicator for a given Skype user name. This macro is now marked as deprecated.
- Template Macro - Allows you to copy the script passed as argument into your own script.
- **Table Of Contents Macro** - Generates a Table Of Content for headings in a document. Currently, we are using the "xwiki/2.0" syntax TOC Macro.
- User Avatar Macro - Displays the avatar of a user or the standard image "noavatar.png" if the user has no avatar.
- **Warning Macro** - Displays a warning message in a styled box format, using a warning icon. This macro is now marked as deprecated and replaced with the newer Warning Macro.
- Yahoo Macro - Displays an online status indicator for the given Yahoo Messenger account id. This macro is now marked as deprecated.

Related Pages

- **Programming Guide**
  - XWiki Scripting
  - XWiki API Reference
  - Velocity Macro
  - User Avatar Macro
  - Template Macro
  - Programming Overview
  - Mime Type Image Macro
  - Livetable Macro
  - Floating Box Macro

Floating Box Macro

The Floating Box macro is written in the "xwiki/1.0" syntax to display a message in a styled floating box format.

Usage

There are 2 ways of using this macro.

```
#floatingbox($text)
#startfloatingbox()
wiki content
#endfloatingbox()
```

Parameters

Name	Optional	Allowed values	Description	Default value
text	No	Any string	The text message that will be displayed in a floating box	None

Examples

```
{{velocity}}
{{html}}
#floatingbox("Table of Contents")
{{/html}}
{{/velocity}}
```

will display


```
{{velocity}}
{{html wiki="true"}}
#startfloatingbox()
This wiki content will be displayed in a styled floating box
* Overview
* What's New
#endfloatingbox()
{{/html}}
{{/velocity}}
```

will display

Related Pages

- **Programming Guide**
  - XWiki Velocity Macros
  - XWiki API Reference
  - User Avatar Macro
  - Template Macro
  - Programming Overview
  - Mime Type Image Macro
  - Livetable Macro

Livetable Macro

The {{livetable /}} macro is mostly used for displaying a table of documents containing an XWiki Object.

Signature

The signature of the {{livetable /}} macro is:

#livetable($id $columns $columnsProperties $options)

- `$id` is a string identifier that allows you to distinguish this table from others and this way it allows you to have several tables on the same page. This id is used in the generated HTML as a prefix for some elements IDs.
- `$columns` is an array that holds the ordered list of columns to display in the table.
- `$columnsProperties` is a hash of properties needed to customize the behavior of each column.
- `$options` is a hash with general options for the table.

The `$columns` Parameter

This parameter allows you to define the livetable columns.

There are 3 major types of columns a table can declare:

- **document columns** displaying and filtering the metadata of the document, such as the author, the last modification date.
- **object properties columns** for a table bound to an XWiki class.
- **special columns** for columns which are not handled by the first two types, such as the list of attachments, the actions that can be performed.

Accepted Values

Name	Description
doc.name	The name of the document.
doc.title	The title of the document.
doc.space	The space of the document.
doc.fullName	The full name of the document.
doc.creationDate	The creation date.
doc.creator	The username of the author.
doc.author	The username of the last author.
doc.date	The date of the last change on the document.
${propertyName}	Any class property the table is bound to. (See the `className` parameter of the `$option` argument for more information on how to bind a table to an XWiki class).
_images	A special column to display all images attached to the retrieved document.
_attachments	A special column to display links to all the attachments of the retrieved document.
_actions	A special column to display a list of actions that can be performed by administrators on the matched documents.
_avatar	A special column to display the user avatar. This works only for a table bound to the "XWiki.XWikiUsers" class.

The `$columnsProperties` Parameter

Accepted Values

Name	Descriptions	Default value
`displayName`	The pretty name of the column header.	*None*
`filterable`	**true** if the column has a filter header.	`true`
`sortable`	**true** if the column is available as a sort key.	`true`

`type`	For filterable columns it represents the filter type and it also allows you to hide a column. The possible values are `hidden`, `text`, `list`, `number`.	*None* (no type)
`size`	The size of the filter field. CSS might override this value to make the field 100%.	*None*
`link`	The type of link to use for the field value. The possible values are: • **"view"** links to the page corresponding to the row • **"field"** links to the page corresponding to the field value (for DBListClass) • **"hidden"** hides the column • **"none"** means no link	*None* (no link)
`html`	**true** if the returned value will be treated as HTML and injected into the row.	`false`
`headerClass`	The name of an optional CSS class to add to the column header of a HTML table.	*None*
`class`	Specifies the full name of the XWiki class for the type of data displayed in the table. This is used by the filtering options of the live table header, when the `$options` hash has a `resultPage` key instead of `className`.	*None*

The `$options` Parameter

Accepted Values

Option name	Description	Default value
className	The full name of the XWiki page holding the class definition for the type of data to display in the table. For instance: "XWiki.XWikiUsers" displays pages containing user objects. If no **className** is given the table will display all the wiki pages.	*None*
resultPage	The full name of the page to use as a JSON data provider for the table. This option allows you to use a data source, other than the default one which is "XWiki.LiveTableResults". This parameter will be ignored if a *className* parameter is present.	*None*
url	This is similar to **resultPage**, except that it accepts an URL instead of the full name of the page it will obtain results from. This option allows you to add extra query parameters. The parameter will be ignored if at least one of the options*className* or *resultPage* is present.	*None*
selectedColumn	The name of the column on which to sort the live table by default. If this option is not present, the first sortable column from the `$columns` array will be used.	*None*
defaultOrder	The default order to sort on the selected column. The accepted values are `asc` and `desc`.	`asc`
rowCount	The maximum number of rows to display in one page of the table.	`15`
maxPages	The maximum number of links to pages displayed in the pagination UI (Not counting the links to the first and last pages that will always be displayed).	`10`
translationPrefix	A string to prefix the table translation keys with (for names of columns for example)	*None*

	in order to have different display messages (translated strings) for different tables.	
tagCloud	Displays a tag cloud filter and an interface which allows users to see entries matching particular tags and the tags matching the current filter selection.	`false`
callback	An advanced option for passing the name of a JavaScript method as custom handler of the matched rows. This method takes care of the row entry DOM construction and of its injection into the table.	*None*
extraParams	Used to add extra parameters to the Ajax request for the *resultPage* which generates the JSON data. Example: *"extraParams" : "page=MySpace.PageName"*.	*None*
topFilters	The HTML fragment that will be placed into a "top filter" area in the same way as the Tag Cloud filter. All filter elements (`input`, `select`) in this fragment will be automatically used as filters for the livetable.	*None*
pageSize	Display a selection box for users to change the number of rows displayed per page in the table.	`true`
pageSizeBounds	Defines the page sizes available in the selection box that allows users to change the number of rows displayed per page in the table. This should be a valid initialization of a JavaScript array of 3 integer values: the minimum page size, the maximum page size, and the steps between proposed page sizes. The default value propose a selection between 10 and 100 with a step of 10.	`[10,100,10]`

To see some examples of usage for the Livetable macro, follow this link.

Related Pages

- **Programming Guide**
  - XWiki Velocity Macros
  - XWiki Query Guide
  - XWiki API Reference
  - User Avatar Macro
  - Template Macro
  - Programming Overview
  - Mime Type Image Macro
  - Floating Box Macro
  - Documents Macro
- **Admin Guide**
  - The App Within Minutes Application

Mime Type Image Macro

The Mime Type Image macro displays an icon for a variety of common mime types and file name extensions. The supported file formats include image formats and Office documents formats.

Usage

#mimetypeimg($mimetype $filename)

Parameters

Name	Optional	Allowed values	Description	Default value
mimetype	No	A string	The mime type of an attachment	None
filename	No	A string	The filename of an attachment	None

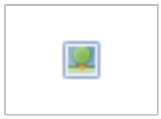
Since the macro does not handle the conversion, both parameters need to be in lowercase.

Examples

For an attached file, you could use the syntax below:

```
{{velocity}}
{{html}}
#mimetypeimg("image/jpeg" "Desert.jpeg")
{{/html}}
{{/velocity}}
```

which will display



Related Pages

- **Programming Guide**
  - XWiki Velocity Macros
  - XWiki API Reference
  - User Avatar Macro
  - Template Macro
  - Programming Overview
  - Livetable Macro
  - Floating Box Macro

Template Macro

The Template macro allows you to include the script passed as argument into the current script. Only the Velocity parser will be executed on the file passed as argument.

Usage

#template($relativeURL)

Only the Velocity parser will be executed on the file passed as argument. The priority of the executed files is as follows:

- a property of the XWiki.XWikiSkins object attached to the current skin document.
- a ".vm" file attached to the current skin document.
- a ".vm" file located in the directory corresponding to the current skin as shown in the below examples.
- the above 3 locations corresponding to the base skin.
- the above 3 locations corresponding to the default skin.
- a ".vm" file located in the "templates" folder as shown in the below examples.

Parameters

The `$relativeURL` is the path to the included script which can be:

- a relative URL to a folder on your filesystem like for instance a skin directory
  #template('C:\Apache Software Foundation\Tomcat 6.0\webapps\xwiki\skins\colibri\customTemplate.vm')
  #template('C:\Apache Software Foundation\Tomcat 6.0\webapps\xwiki\skins\colibri\templates\customTemplate.vm')
- the name of a default XWiki Velocity template file (i.e. located in the "templates" folder)
  #template('warning.vm')

Related Pages

- **Programming Guide**
  - XWiki Velocity Macros
  - XWiki API Reference
  - User Avatar Macro
  - Programming Overview
  - Mime Type Image Macro
  - Livetable Macro
  - Floating Box Macro

User Avatar Macro

The User Avatar macro displays the avatar of a user or the standard image "noavatar.png" in case the user has no avatar.

Usage

#useravatar($username)

Parameters

Name	Optional	Allowed values	Description	Default value
username	No	A string	An XWiki user page	None

Examples

{{velocity}}
{{html}}
#useravatar("xwiki:XWiki.Admin")
{{/html}}
{{/velocity}}

will display

 or 

Related Pages

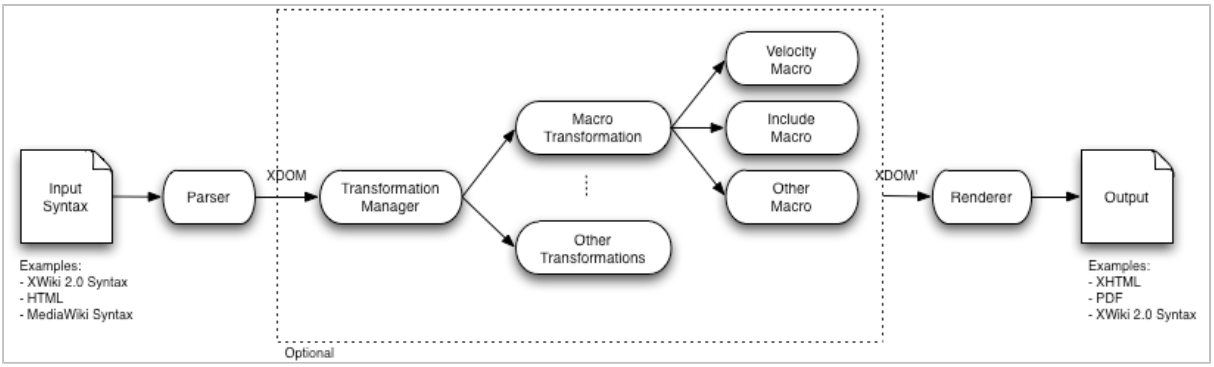
- **User Guide**
  - User Profile
- **Programming Guide**
  - XWiki Velocity Macros
  - XWiki API Reference
  - Template Macro
  - Programming Overview
  - Mime Type Image Macro
  - Livetable Macro
  - Floating Box Macro

XWiki Rendering Macros in Java

XWiki Rendering Macros in Java are created using the XWiki Rendering architecture and Java. In order to implement a Java macro you will need to write 2 classes:

- One that representing the allowed parameters, including mandatory parameters, default values, parameter descriptions. An instance of this class will be automatically populated when the user calls the macro in wiki syntax.
- Another one that is the macro itself and implements the `Macro` interface.

The XWiki Rendering Architecture can be summarized by the image below:



The Parser parses a text input like "xwiki/2.0" syntax or HTML and generates an XDOM object. This object is an Abstract Syntax Tree which represents the input into structured blocks.

The Renderer takes an XDOM as input and generates an output like "xwiki/2.0" syntax, XHTML or PDF.

The Transformation takes an XDOM as input and generates a modified one. The available Transformations are:

- Macro Transformation explained below
- Icon Transformation that transforms series of characters into icons (emoticons and others).
- Link Checker Transformation that checks the validity of external links.
- WikiWord Transformation that automatically generates page links for WikiWords.

The Macro Transformation searches for all macro blocks inside the XDOM object and replaces them with blocks generated by various macros. For each macro block found, the Macro Transformation verifies whether there is a corresponding macro registered for it and for the target syntax. Macros can be registered for all syntaxes or only for a given syntax. Finally, macros are sorted by priority and executed one by one by replacing the macro block with the list of blocks generated by the macro. The process is repeated until there are no more macro blocks left in the XDOM.

The available macros in the XWiki Rendering modules are:

- Box Macro
- Comment Macro
- Content Macro
- Footnote Macro
- HTML Macro
- ID Macro
- JIRA Macro
- Info Message Macro
- Warning Message Macro
- Error Message Macro
- Table of Contents Macro

The steps to follow for creating and deploying an XWiki Rendering macro in Java are detailed here.

Java Macros Bundled with XWiki Enterprise

- Box Macro - Renders wiki content in a graphical box which can contain a title and an image.
- Cache Macro - Caches the content of the macro for a determined time.
- Chart Macro - Generates graphical charts from various data sources.
- Code Macro - Used to highlight code.
- Comment Macro - Adds comments.

- Container Macro - Lays out wiki content.
- Content Macro - Allows adding content in any of the supported syntaxes.
- Context Macro - Executes content with another document set as the current document.
- Dashboard Macro - Displays a set of gadgets in a dashboard.
- Display Macro - Used to display an entity (document, property, etc.)
- Error Message Macro - Displays an error message in a styled box format, with an error icon.
- Footnote Macro - Adds a note of text placed by default at the bottom of the page.
- Formula Macro - Inserts a mathematical formula.
- Gallery Macro - Displays an image gallery.
- Groovy Macro - Adds the ability to write Groovy scripts in wiki pages.
- HTML Macro - Allows inserting HTML or XHTML in wiki pages.
- Id Macro - Allows placing a reference/location in a page.
- Include Macro - Allows the inclusion of other pages into the current page.
- Info Message Macro - Displays an information message in a styled box format, using an information icon.
- Office Macro - Displays the content of an Office attachment as HTML inside the current document.
- Put Footnotes Macro - Displays the footnotes mentioned so far in the wiki code and resets the footnote counter.
- Python Macro - Adds the ability to write Python scripts in wiki pages.
- RSS Macro - Displays the contents of a RSS feed.
- Script Macro - Executes a script implementing the JSR-223 API.
- Success Message Macro - Displays a succes message in a styled box format.
- TOC Macro - Generates a Table Of Content for headings into a document.
- Translation Macro - Allows you to insert a translation message in the page content.
- User Avatar Macro - Displays the avatar of a user, or a standard image **noavatar.png** if the user has no avatar.
- Velocity Macro - Executes a Velocity script and applies the current page parser on the result.
- Warning Message Macro - Displays a warning message in a styled box format, using a warning icon.

Related Pages

- **Programming Guide**
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro

Box Macro

The Box macro renders wiki content in a graphical box which can contain a title and an image.

Usage

```
{{box title="<title>" image="<image>" cssClass="<cssClass>" /}}
```

Parameters

Name	Optional	Allowed values	Description	Default value
title	Yes	A string: raw text or "xwiki/2.0" syntax	The title displayed in the message box	None
image	Yes	A string representing an absolute URL	The image to display in the message box	None
cssClass	Yes	A string	The CSS sheet to use for rendering the box	None

Example

```
{{box title="Content" image="http://mywiki.com/xwiki/bin/download/ColorThemes/DefaultColorTheme/logo.png"
cssClass="floatingBox"}}
* Overview
* What's New
{{/box}}
```

will display

The custom CSS can be included in a StyleSheetExtension object attached to the page. For the above example the CSS rules for the ".floatingBox" class are:

```
.floatingBox img{
width: 50px;
}
.floatingBox{
width: 500px;
}
```

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro

Cache Macro

The Cache macro caches the content of a page for a predefined period of time.

Usage

```
{{cache id="<id>" timeToLive="<number of seconds>" maxEntries="<number of entries>"}}
Content to cache
{{/cache}}
```

Parameters

Name	Optional	Allowed values	Description	Default value
id	Yes	A string	The identifier used as the key to the cache. In case you might need to generate it using script macros, you can use XWiki syntax. Since the same content can exist on several pages with different results, the uniqueness of the id is not guaranteed.	If not defined, the Cache macro uses its own content as id.
timeToLive	Yes	A number	The cache duration in seconds which actually represents the number of seconds after which the cache will expire, provided no user accesses the page. The maximum value is 2147483647 seconds.	300 (seconds)
maxEntries	Yes	A number	The maximum number of cached entries in a given cache.	1000

Example

```
{{cache id="planningPage" timeToLive="2502000"}}
This page won't change for a while, so there is no need to reload it each time.
{{/cache}}
```

To compute the id based on a script evaluation, you could use the code below:

```
{{velocity}}
#set ($myid="myComputedId")
{{/velocity}}
{{cache id="~{~{velocity~}~}$myid~{~{/velocity~}~}"}}
This page won't change for a while, so there is no need to reload it each time.
{{/cache}}
OR
{{cache id="~{~{velocity~}~}$request.id~{~{/velocity~}~}"}}
This page won't change for a while, so there is no need to reload it each time.
{{/cache}}
```

When using script macros in the content of the Cache macro, then these scripts will be executed only once, before they are cached.

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference

Chart Macro

The Chart macro displays charts from given data sources.

Usage

{{chart source="<inline|xdom>" type="<type>" params="<params>" title="<title>" width="<width>" height="<height>" colors="<colors>" /}}

Parameters

Name	Optional	Allowed values	Description	Default value
source	Yes	• xdom • inline	The type of data source providing input for charting	`inline` if the macro has content, `xdom` otherwise
type	No	• line • area • bar • pie • bar3D • line3D • xy_area • xy-line-and-shape • xy_line3D • xy_step	The type of the generated chart	None
params	Yes	A string	A list of parameter name to value mappings, depending on the data source	None
title	Yes	A string	The title used when rendering the chart	None
width	Yes	A string	The width of the chart image in pixels	400
height	Yes	A string	The height of the chart image in pixels	300
colors	Yes	A string	The list of colors to use The values are specified in hexadecimal.	None

The `params` parameter

The "params" parameter can contain the following mappings:

- `document`: the name of the wiki page
- `table`: the id of the table source
- `time_period`
  - millisecond
  - second
  - minute
  - hour
  - day
  - week
  - month
  - quarter
  - year
- `locale`
- `date_format`
- axis related mappings:
  - `domain_axis_type`: the name of the domain axis parameter (e.g. `date`)

    - `range_axis_type`: the name of the range axis parameter
    - `domain_axis_date_format`: the name of the domain axis date format parameter that should only be used if the axis type is date
    - `range_axis_date_format`: the name of the range axis date format parameter that should only be used if the axis type is date
    - `domain_axis_lower`: the lower limit on the domain axis
    - `domain_axis_upper`: the upper limit on the domain axis
    - `range_axis_lower`: the lower limit on the range axis
    - `range_axis_upper`: the upper limit on the range axis
- `range`: the data range selected just like for an Excel sheet
- `series`
    - `columns` which defines the **x** and **y** axes
    - `rows`
- `dataset`
    - `category`
    - `timeseries`
    - `pie`
    - `xy`
    - `timetable_xy`

Examples

Generate a Bar chart from an `xdom` data source

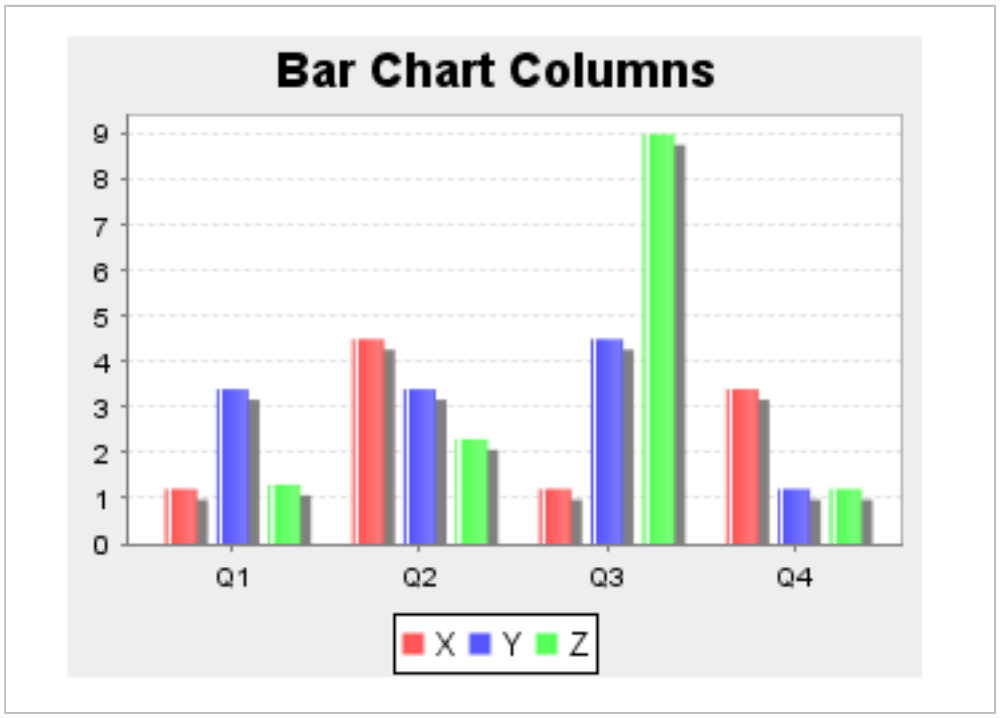
```
{{chart type="bar" params="document:Sandbox.Test;table:barChartTable;range:B2-D5;series:columns;" title="Bar Chart Columns" width="320" height="240"/}}
```

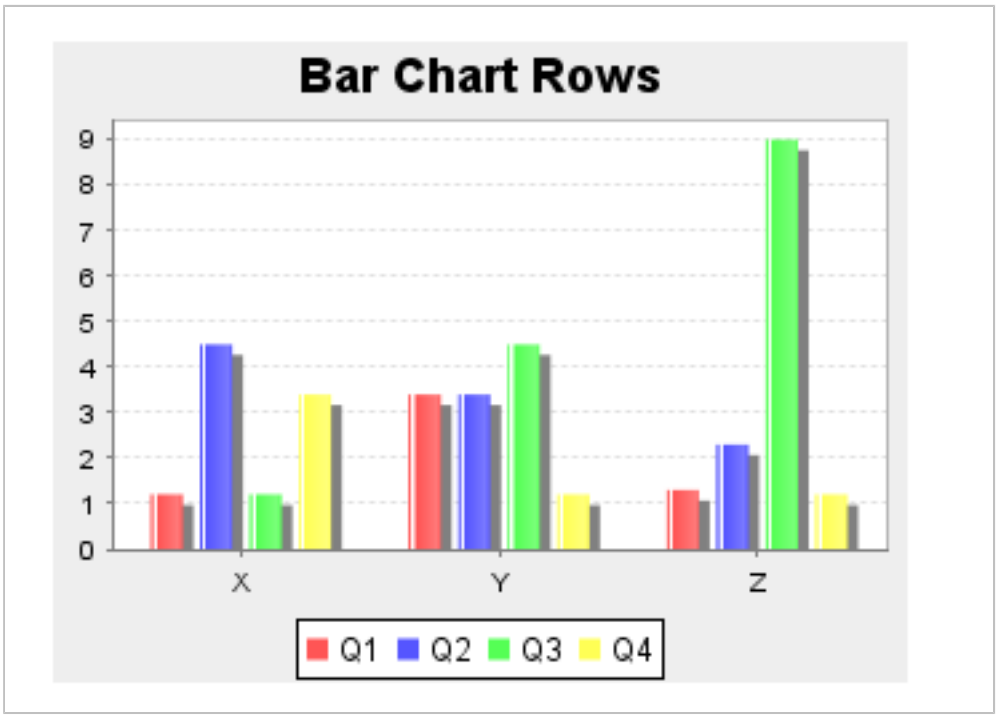
and

```
{{chart type="bar" params="document:Sandbox.Test;table:barChartTable;range:B2-D5;series:rows;" title="Bar Chart Rows" width="320" height="240"/}}
```

where the page "Sandbox.Test" contains the following code:

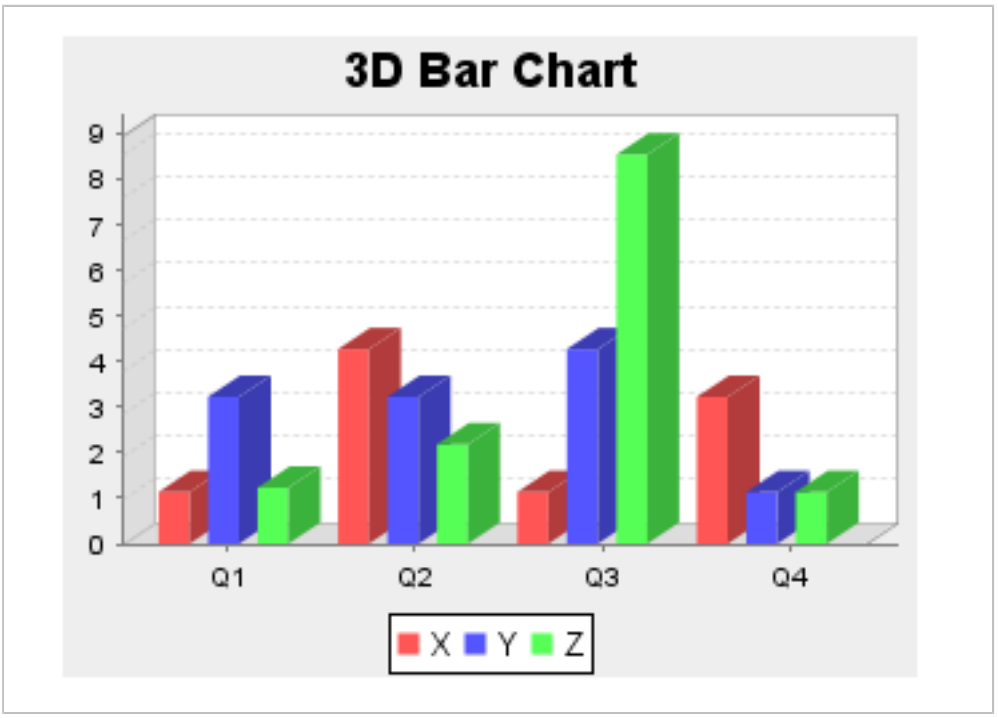
```
(%id="barChartTable"%)
|=|=X|=Y|=Z
|Q1|1.2|3.4|1.3
|Q2|4.5|3.4|2.3
|Q3|1.2|4.5|9.0
|Q4|3.4|1.2|1.2
```





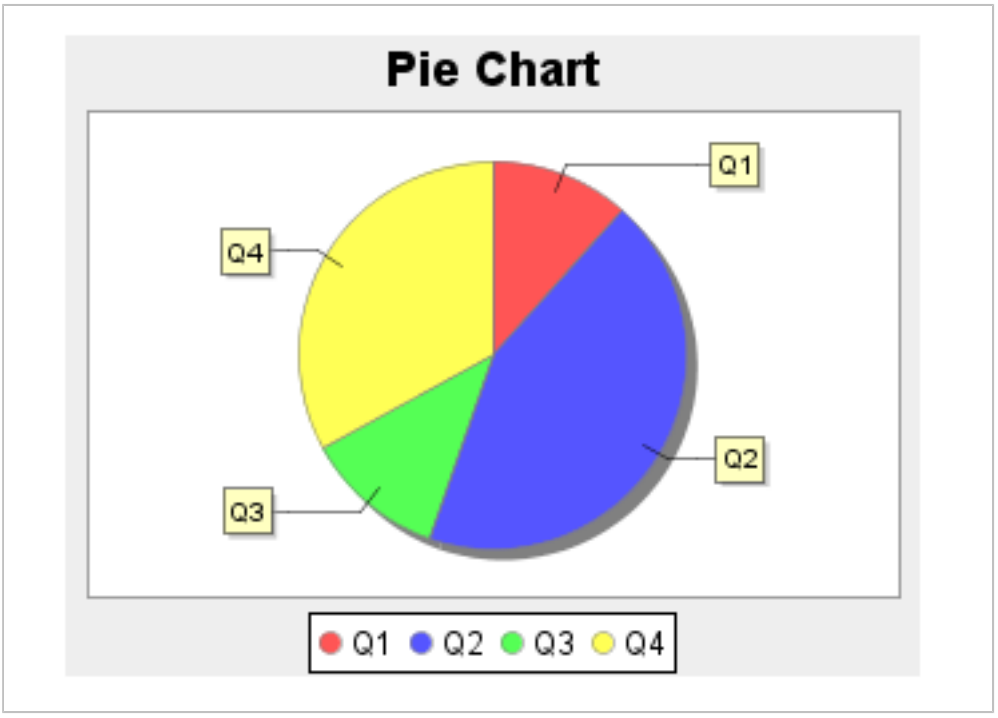
Generate a 3D Bar chart from an `inline` data source

```
{{chart type="bar3D" source="inline" params="range:B2-D5;series:columns;" title="3D Bar Chart" width="320" height="240"}}
|=|=X|=Y|=Z
|Q1|1.2|3.4|1.3
|Q2|4.5|3.4|2.3
|Q3|1.2|4.5|9.0
|Q4|3.4|1.2|1.2
{{/chart}}
```



Generate a Pie chart from an `inline` data source

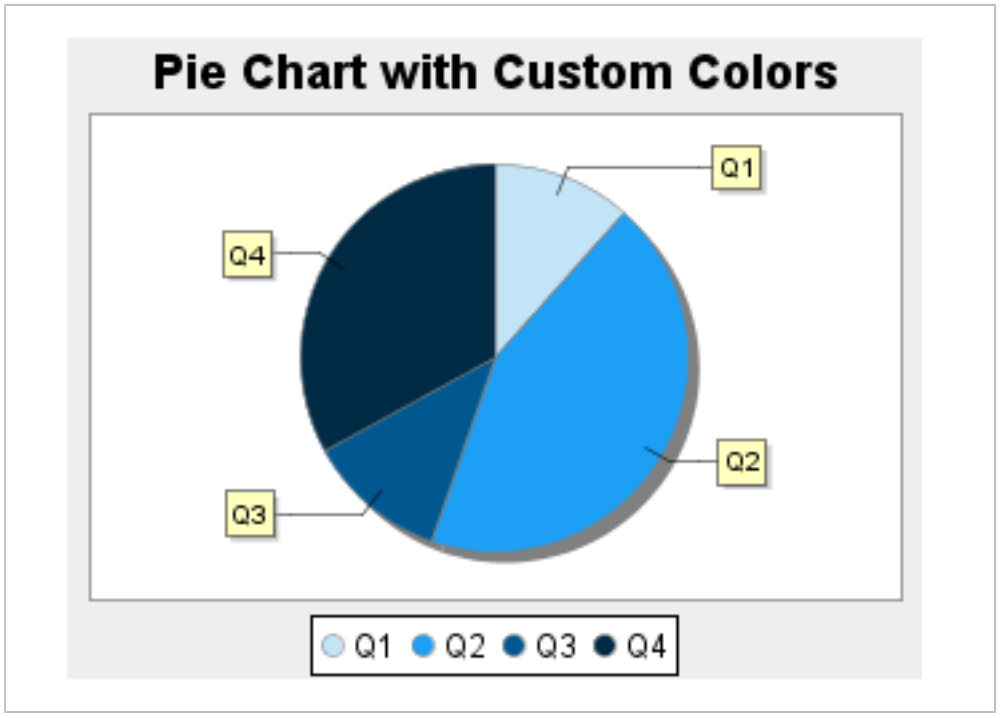
```
{{chart type="pie" source="inline" params="range:B2-D5;series:columns;" title="Pie Chart" width="320" height="240"}}
|=|=X|=Y|=Z
|Q1|1.2|3.4|1.3
|Q2|4.5|3.4|2.3
|Q3|1.2|4.5|9.0
|Q4|3.4|1.2|1.2
{{/chart}}
```



Generate a Pie chart with custom colors

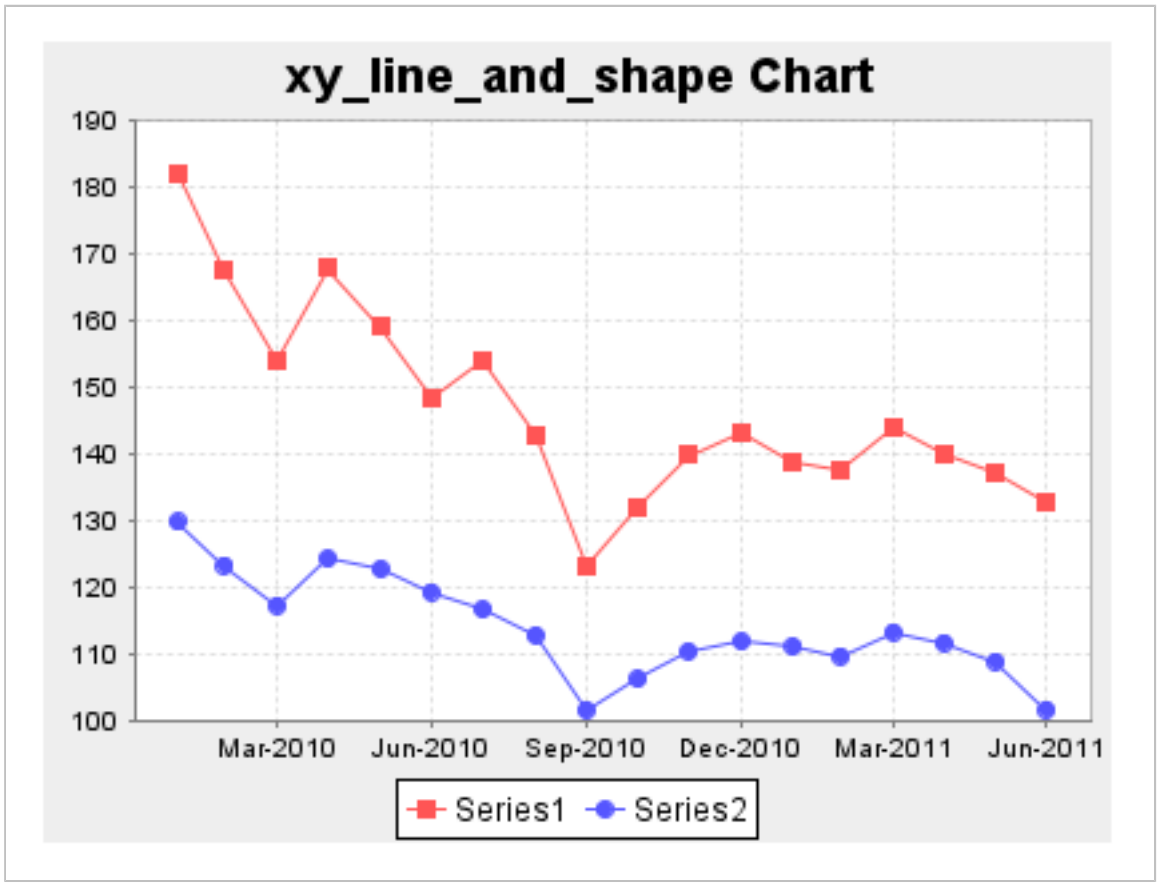
```
{{chart type="pie" source="inline" params="range:B2-D5;series:columns;colors:C3E3F7,1D9FF5,015891,012A45" title="Pie Chart with Custom Colors" width="320" height="240"}}
```

```
|=|=X|=Y|=Z
|Q1|1.2|3.4|1.3
|Q2|4.5|3.4|2.3
|Q3|1.2|4.5|9.0
|Q4|3.4|1.2|1.2
{{/chart}}
```



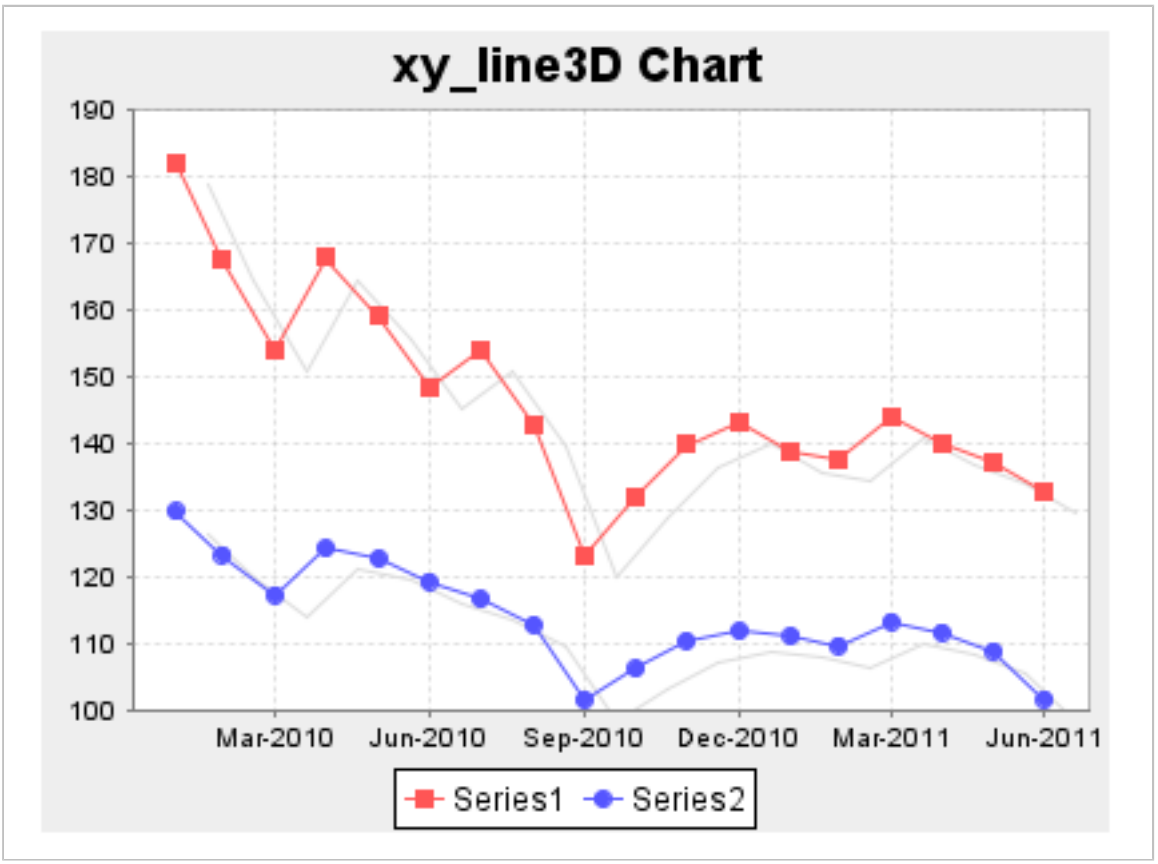
Generate a "xy_line_and_shape" chart

```
{{chart type="xy_line_and_shape" params="range:B2-C19;dataset:timetable_xy;domain_axis_type:date;domain_axis_date_format:MMM-yyyy;date_format:yyyy-MM;time_period:month;range_axis_lower:100;range_axis_upper:190" title="xy_line_and_shape Chart"}}
|=|=Series1|=Series2
|2010-2|181.8|129.6
|2010-3|167.3|123.2
|2010-4|153.8|117.2
|2010-5|167.6|124.1
|2010-6|158.8|122.6
|2010-7|148.3|119.2
|2010-8|153.9|116.5
|2010-9|142.7|112.7
|2010-10|123.2|101.5
|2010-11|131.8|106.1
|2010-12|139.6|110.3
|2011-1|142.9|111.7
|2011-2|138.7|111.0
|2011-3|137.3|109.6
|2011-4|143.9|113.2
|2011-5|139.8|111.6
|2011-6|137.0|108.8
|2011-7|132.8|101.6
{{/chart}}
```



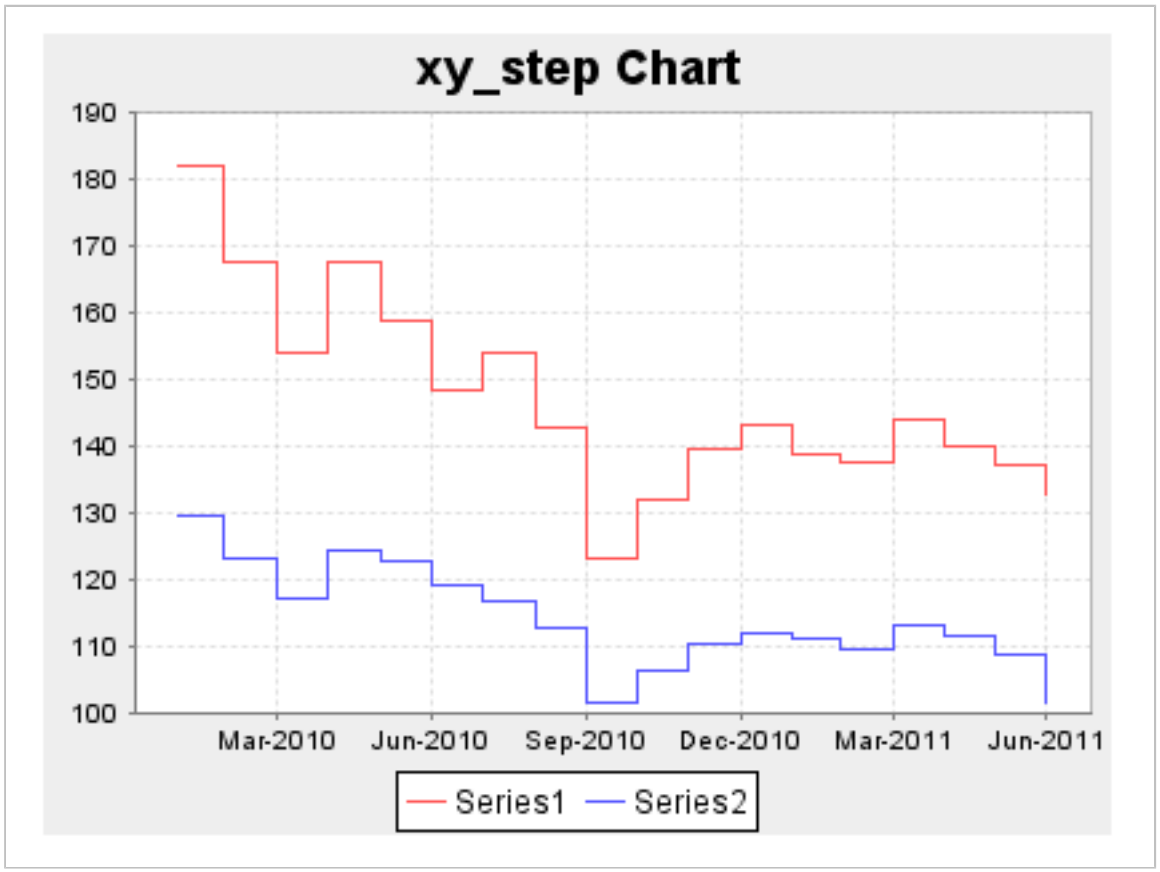
Generate a "xy_line3D" chart

```
{{chart type="xy_line3D" params="range:B2-C19;dataset:timetable_xy;domain_axis_type:date;domain_axis_date_format:MMM-yyyy;date_format:yyyy-MM;time_period:month;range_axis_lower:100;range_axis_upper:190" title="xy_line3D Chart"}}
|=|=Series1|=Series2
|2010-2|181.8|129.6
|2010-3|167.3|123.2
|2010-4|153.8|117.2
|2010-5|167.6|124.1
|2010-6|158.8|122.6
|2010-7|148.3|119.2
|2010-8|153.9|116.5
|2010-9|142.7|112.7
|2010-10|123.2|101.5
|2010-11|131.8|106.1
|2010-12|139.6|110.3
|2011-1|142.9|111.7
|2011-2|138.7|111.0
|2011-3|137.3|109.6
|2011-4|143.9|113.2
|2011-5|139.8|111.6
|2011-6|137.0|108.8
|2011-7|132.8|101.6
{{/chart}}
```



Generate a "xy_step" chart

```
{{chart type="xy_step" params="range:B2-C19;dataset:timetable_xy;domain_axis_type:date;domain_axis_date_format:MMM-yyyy;date_format:yyyy-MM;time_period:month;range_axis_lower:100;range_axis_upper:190" title="xy_step Chart"}}
|=|=Series1|=Series2
|2010-2|181.8|129.6
|2010-3|167.3|123.2
|2010-4|153.8|117.2
|2010-5|167.6|124.1
|2010-6|158.8|122.6
|2010-7|148.3|119.2
|2010-8|153.9|116.5
|2010-9|142.7|112.7
|2010-10|123.2|101.5
|2010-11|131.8|106.1
|2010-12|139.6|110.3
|2011-1|142.9|111.7
|2011-2|138.7|111.0
|2011-3|137.3|109.6
|2011-4|143.9|113.2
|2011-5|139.8|111.6
|2011-6|137.0|108.8
|2011-7|132.8|101.6
{{/chart}}
```



Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro

- Code Macro
- Charting Plugin
- Cache Macro
- Box Macro

Code Macro

The Code macro is used to highlight code in languages supported by Pygments.

Usage

```
{{code language=<language>}}
// Lines of code
{{/code}}
```

Parameters

Name	Optional	Allowed values	Description	Default value
language	Yes	A string	The language identifier for the code snippet	If not specified, the macro will try to determine the syntax. To avoid the highlighting, you can use `none`.

The Code macro also supports all the parameters of the Box macro which are:

Name	Optional	Allowed values	Description	Default value
title	Yes	A string: raw text or "xwiki/2.0" syntax	The title displayed in the message box	None
image	Yes	A string representing an absolute URL	The image to display in the message box	None
cssClass	Yes	A string	The CSS sheet to use for rendering the box	None

Customization

It is possible to configure the style used by the Code macro via the "WEB-INF/xwiki.properties" file by adding the parameter:

rendering.macro.code.pygments.style

The default value is `default` but you can actually specify any style name supported by Pygments.

Examples

Using only the `language` parameter

```
{{code language="java"}}
{{groovy}}
return java.util.Arrays.asList(new org.xwiki.rendering.block.WordBlock("Hello"),
org.xwiki.rendering.block.SpaceBlock.SPACE_BLOCK, new org.xwiki.rendering.block.WordBlock("world"));
{{/groovy}}
{{/code}}
```

will display

```
{{groovy}}
return java.util.Arrays.asList(new org.xwiki.rendering.block.WordBlock("Hello"),
org.xwiki.rendering.block.SpaceBlock.SPACE_BLOCK, new org.xwiki.rendering.block.WordBlock("world"));
{{/groovy}}
```

Using the Box macro parameters

```
{{code language="java" title="Content" image="http://mywiki.com/xwiki/bin/download/Main/WebHome/Tulips.jpg"
cssClass="floatingBox"}}
{{groovy}}
```

```
return java.util.Arrays.asList(new org.xwiki.rendering.block.WordBlock("Hello"),
org.xwiki.rendering.block.SpaceBlock.SPACE_BLOCK, new org.xwiki.rendering.block.WordBlock("world"));
{{/groovy}}
{{/code}}
```

will display

The custom CSS can be included in a StyleSheetExtension object attached to the page. For the above example the CSS rules for the ".floatingBox" class are:

```
.floatingBox img{
width: 50px;
}
.floatingBox{
width: 800px;
}
```

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Comment Macro

The Comment macro allows adding comments that will not be visible for other users.

Usage

```
{{comment}}
// Text to display in the comment
{{/comment}}
```

Example

```
{{comment}}
Get a QueryManager injected.
{{/comment}}
{{code language="java"}}
@Inject
@Named("secure")
private QueryManager queryManager;
{{/code}}
```

will display

```
@Inject
@Named("secure")
private QueryManager queryManager;
```

Related Pages

- **User Guide**
  - Page Comments
  - Annotations Application
- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - The "xwikicomments" Table
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro

    - Container Macro
    - Code Macro
    - Chart Macro
    - Cache Macro
    - Box Macro
- **Admin Guide**
    - Configure Annotations

Container Macro

The Container macro displays wiki content in columns.

Usage

The Container macro is in fact used to define the layout. The columns will be delimited by the Group syntactic elements: `(((...)))`.

```
{{container layoutStyle="<columns|any string>" justify="<true|false>"}}
(((first column)))
(((second column)))
{{/container}}
```

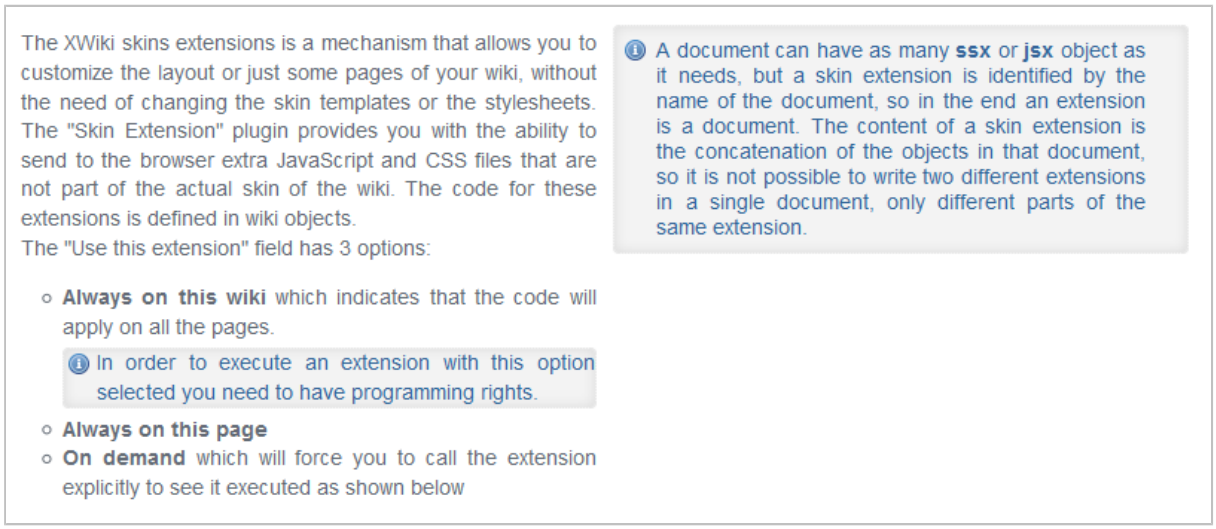
Parameters

Name	Optional	Allowed values	Description	Default value
layoutStyle	Yes	The `columns` layout is implemented by default, but you can use any string	The layout used for the content grouped by the macro	None. If no value is specified, the content will be rendered as if it wasn't using the macro (i.e. no value is applied).
justify	Yes	• `true` • `false`	A flag stating whether the content in the container is justified or not.	false

Example

```
{{container layoutStyle="columns" justify="true"}}
(((The XWiki skins extensions is a mechanism that allows you to customize the layout or just some pages of your wiki, without the need of changing the skin templates or the stylesheets. The "Skin Extension" plugin provides you with the ability to send to the browser extra JavaScript and CSS files that are not part of the actual skin of the wiki. The code for these extensions is defined in wiki objects.
The "Use this extension" field has 3 options:
* **Always on this wiki** which indicates that the code will apply on all the pages. {{info}}In order to execute an extension with this option selected you need to have programming rights.{{/info}}
* **Always on this page**
* **On demand** which will force you to call the extension explicitly to see it executed as shown below)))
((({{info}}
A document can have as many **ssx** or **jsx** object as it needs, but a skin extension is identified by the name of the document, so in the end an extension is a document. The content of a skin extension is the concatenation of the objects in that document, so it is not possible to write two different extensions in a single document, only different parts of the same extension.
{{/info}})))
{{/container}}
```

will display:



Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Content Macro

The Content macro allows you to enter content in any of the supported syntaxes: `xwiki/2.0, xwiki/2.1, confluence/1.0, jspwiki/1.0, creole/1.0, mediawiki/1.0, twiki/1.0, xhtml/1.0, html/4.01, plain/1.0, docbook/4.4, markdown/1.0, markdown/1.1, apt/1.0`.

The list of supported syntaxes can be configured via the "WEB-INF/xwiki.cfg" file by editing the `xwiki.rendering.syntaxes` parameter. The default value is `xwiki/2.0, xwiki/2.1`.

Usage

```
{{content syntax="<syntax>"}}
// Content in the chosen syntax
{{/content}}
```

Parameters

Name	Optional	Allowed values	Description	Default value
syntax	No	xwiki/2.0, xwiki/2.1, confluence/1.0, jspwiki/1.0, creole/1.0, mediawiki/1.0, twiki/1.0, xhtml/1.0, html/4.01, plain/1.0, docbook/4.4, markdown/1.0, markdown/1.1, apt/1.0	The technical id of the syntax in which the content is written	None

Examples

xwiki/2.0 syntax

```
{{content syntax="xwiki/2.0"}}
[[This is an "xwiki/2.0" link>>xwiki:Sandbox.WebHome#HHeadings]]
{{/content}}
```

will display

This is an "xwiki/2.0" link

xwiki/2.1 syntax

```
{{content syntax="xwiki/2.1"}}
[[This is an "xwiki/2.1" link>>xwiki:Sandbox.WebHome||anchor="HHeadings]]
{{/content}}
```

will display

This is an "xwiki/2.1" link

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro

Context Macro

The Context macro executes content with another document set as the current document. The most common use case is the need to display content from other pages in the current document but, at the same time, the other document has to handle the links and images.

Usage

```
{{context document="<documentName>"}}
// Content in XWiki syntax
{{/context}}
```

Parameters

Name	Optional	Allowed values	Description	Default value
document	No	A string	The name of the wiki page whose context will be used as the current context (full name or prefixed full name)	None

Example

```
{{context document="xwiki:XWiki.DefaultSkin"}}
{{velocity}}
#set($obj = $doc.getObject("XWiki.XWikiSkins", 0))
$doc.display("name", $obj)
{{/velocity}}
{{/context}}
```

will display

Default XWiki Skin

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro

Dashboard Macro

The Dashboard macro creates a dashboard page in which to display a collection of gadgets grouped in columns.

Usage

To create an empty dashboard page, just edit the page in "Wiki" mode and use the following syntax:

```
{{dashboard style="<style>" layout="<column>" source="<source>" /}}
```

Parameters

Name	Optional	Allowed values	Description	Default value
style	Yes	Any string	The style of the gadgets on the dashboard	Empty string. A predefined value is `panels` which will display the gadgets like panels. You can also define your own style using stylesheet extensions.
layout	Yes	Any string but only `columns` is implemented by default	The layout of the gadgets on the dashboard	None
source	Yes	A string representing the name of a document	The page from where the objects defining the dashboard gadgets should be read	None

Examples

To know more about how to add and customize gadgets, you could read the following tutorials listed in the "Related Pages" section:

- Create and customize the main home page dashboard in the "Dashboard Application" documentation page
- Create and customize an user dashboard in the "Dashboard Application" documentation page
- Create and customize a space dashboard in the "Create and Delete a Space" documentation page

Related Pages

- **User Guide**
  - XWiki Dashboard Application
  - Create and Delete a Space
- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro

Display Macro

The Display macro allows displaying an entity, document or object, in its own context.

Usage

{{display reference="<reference>" type="<document>" section="H<section>"/}}

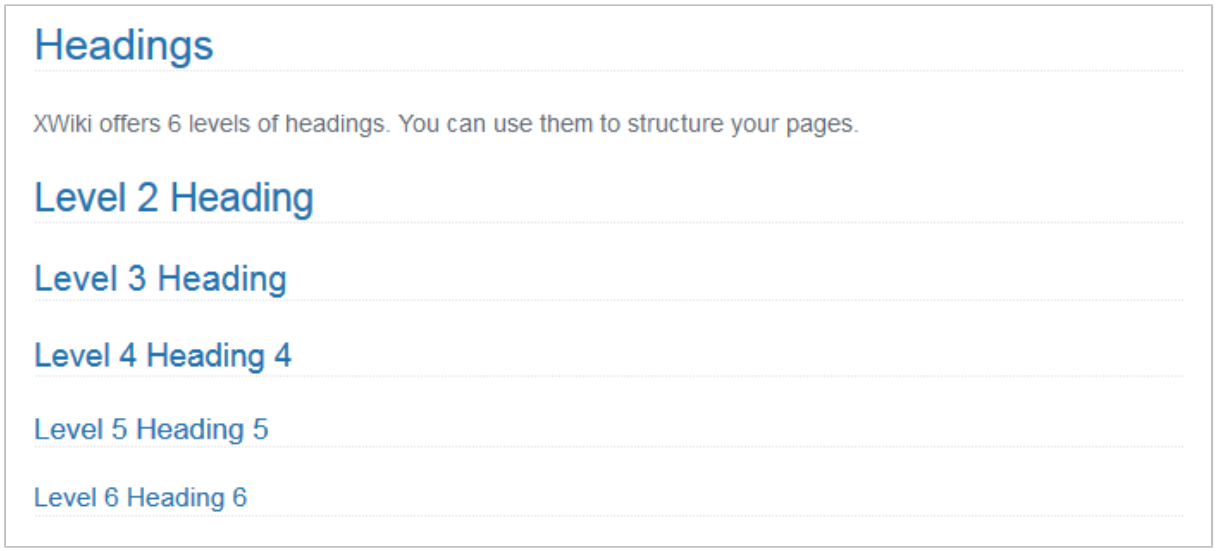
Parameters

Name	Optional	Allowed values	Description	Default value
reference	No	An entity reference	The name of the entity (document, object) to display	None
type	Yes	`document`	The type of the entity	`document`
section	Yes	A string	An anchor to the displayed section of the target reference, having the form: "HSectionName"	None

Example

{{display reference="xwiki:Sandbox.WebHome" type="document" section="HHeadings" /}}

will display



Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro

Error Macro

The Error Message macro displays an error message in a styled box format.

Usage

{{error}}Your custom message.{{/error}}

To learn how to further customize the box format, go to the Notification Widget Tutorial.

Example

{{error}}
The server will be restarted today at 7PM and the wiki might be unavailable for a few minutes.
{{/error}}

will display

The server will be restarted today at 7PM and the wiki might be unavailable for a few minutes.

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Footnote Macro

The Footnote macro adds a note of text to the bottom of the page. The note of text is marked by a superscript number containing a link to the area in the page where the footnote is displayed. On the other hand, the text in the footnote area is preceded by a backlink to the area in the text where it is referred.

Usage

{{footnote}}
// Text to display in the footnote
{{/footnote}}

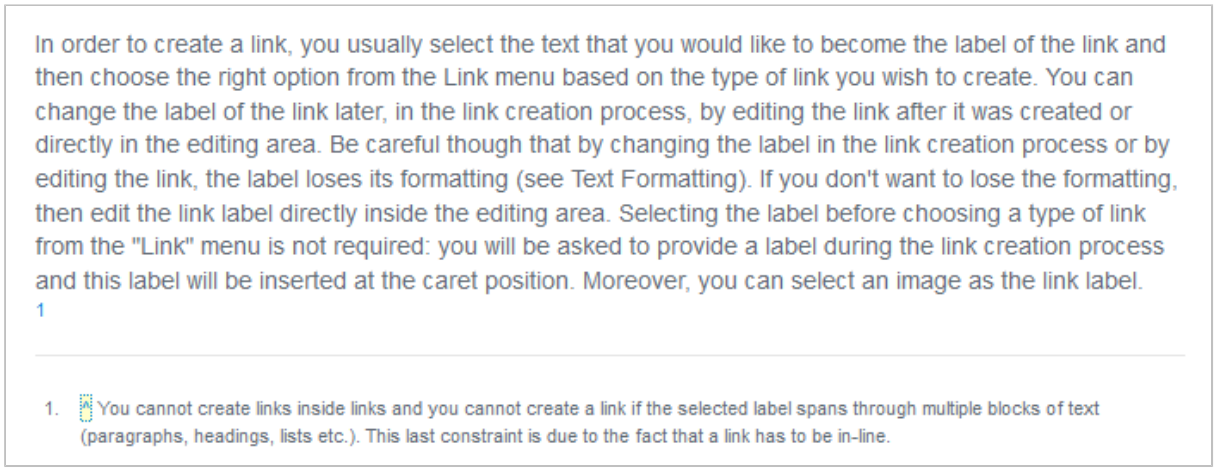
Example

In order to create a link, you usually select the text that you would like to become the label of the link and then choose the right option from the Link menu based on the type of link you wish to create. You can change the label of the link later, in the link creation process, by editing the link after it was created or directly in the editing area. Be careful though that by changing the label in the link creation process or by editing the link, the label loses its formatting (see Text Formatting). If you don't want to lose the formatting, then edit the link label directly inside the editing area. Selecting the label before choosing a type of link from the "Link" menu is not required: you will be asked to provide a label during the link creation process and this label will be inserted at the caret position. Moreover, you can select an image as the link label.
{{footnote}}
You cannot create links inside links and you cannot create a link if the selected label spans through multiple blocks of text (paragraphs, headings, lists etc.). This last constraint is due to the fact that a link has to be in-line.
{{/footnote}}

will display

In order to create a link, you usually select the text that you would like to become the label of the link and then choose the right option from the Link menu based on the type of link you wish to create. You can change the label of the link later, in the link creation process, by editing the link after it was created or directly in the editing area. Be careful though that by changing the label in the link creation process or by editing the link, the label loses its formatting (see Text Formatting). If you don't want to lose the formatting, then edit the link label directly inside the editing area. Selecting the label before choosing a type of link from the "Link" menu is not required: you will be asked to provide a label during the link creation process and this label will be inserted at the caret position. Moreover, you can select an image as the link label. [1]

When the superscript number link is clicked, the corresponding footnote is highlighted.



Also, when clicking the link that leads back to the reference of a footnote, the corresponding footnote marker is highlighted.

1

1. ^ You cannot create links inside links and you cannot create a link if the selected label spans through multiple blocks of text (paragraphs, headings, lists etc.). This last constraint is due to the fact that a link has to be in-line.

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Formula Macro

The Formula macro renders as image a mathematical formula written in TeX markup.

Usage

```
{{formula imageType="<png|gif|jpg>" fontSize="<fontSize>"}}
//LaTeX mathematical formula
{{/formula}}
```

Parameters

Name	Optional	Allowed values	LaTeX correspondent	Description	Default value
imageType	Yes	• png • gif • jpg		The image format	png
fontSize	Yes	• tiny • very_small • smaller • small • normal • large • larger • very_large • huge • extremely_huge	• tiny • scriptsize • footnotesize • small • normalsize • large • Large • LARGE • huge • Huge	Font sizes corresponding to LaTeX font sizes	normal

Examples

```
{{formula}}
\begin{math}\int\frac
{d\theta}{1+\theta^2} = \tan^{-1}
\theta+ C\end{math}
{{/formula}}
```

will display

$$\int \frac{d\theta}{1+\theta^2} = \tan^{-1}\theta + C$$

```
{{formula}}
\begin{align}
\cos^2\psi+\sin^2\psi & = 1, \\
\cosh^2\omega-\sinh^2\omega &=1.
\end{align}
{{/formula}}
```

will display

$$\cos^2\psi + \sin^2\psi = 1, \cosh^2\omega - \sinh^2\omega = 1.$$

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference

Gallery Macro

The Gallery macro displays an image gallery using attached files or images located at external URLs.

Usage

{{gallery}}
image://relative path to attached image//
OR
image://link to external URL
{{/gallery}}

Examples

Display the images attached to "Main.WebHome"

{{gallery}}
image:xwiki:Main.WebHome@Tulips.jpg
image:xwiki:Main.WebHome@Desert.jpg
{{/gallery}}

will display

Display all images visible on a page

You can display the images visible on another page by using the Display macro inside the Gallery macro.

{{gallery}}
{{display reference="Main.Team"/}}
{{/gallery}}

Display images from an RSS feed

You can display the images from an RSS feed by using the RSS macro inside the Gallery macro.

{{gallery}}

```
{{rss feed="http://api.flickr.com/services/feeds/photos_public.gne?format=rss_200&tags=xwiki" content="true"
decoration="false" /}}
{{/gallery}}
```

Display images attached to the current page

To do so, you can use the Attachment Gallery macro which extends the Gallery macro.

Keyboard Shortcuts

Key	Functionality
←	Display the previous image
→	Display the next image
Home	Display the first image
End	Display the last image
F	Toggle full-screen view
Esc	Exit full-screen view

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Groovy Macro

The Groovy macro adds the ability to write Groovy scripts in wiki pages. This is a shortcut for Script Macro equivalent to `{{script language="groovy"}}` to execute Groovy scripts.

Usage

```
{{groovy}}
def list = ["one", "two"]
list.each { item ->
  println "* ${item}"
}
{{/groovy}}
```

Parameters Definition

Name	Optional	Allowed values	Default value	Description
language	yes	the identifier of the JSR 223 engine		Indicates which engine to use to execute the provided script.
output	yes	true/false	true	Indicates the output result has to be inserted back in the document.
wiki	yes	true/false	true	indicates if the result of the script execution has to be parsed by the current wiki parser. If not, it is put into a verbatim block.
jars	yes	comma-separated list of JARs that will be added to the script execution class loader	none	

Example

```
{{groovy}}
def list = ["one", "two"]
list.each { item ->
  println "* ${item}"
}
{{/groovy}}
```

Result

- one
- two

Note that since **context** is a reserved binding in JSR-223 specifications used for Scripting Macros, the **XWiki API Context** is now accessible through **xcontext**.

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro

HTML Macro

The HTML macro allows you to insert HTML and XHTML content in wiki pages.

Usage

```
{{html wiki="<true|false>" clean="<true|false>"}}
// HTML content
{{/html}}
```

Parameters

Name	Optional	Allowed values	Description	Default value
wiki	Yes	true false	A flag stating whether the wiki syntax inside the HTML macro will be evaluated or not	false
clean	Yes	true false	A flag stating whether the user input should be converted to valid XHTML or not	true

Examples

The wiki syntax inside the macro is not interpreted

```
{{html wiki="false" clean="false"}}
{{warning}}This is a warning message.{{/warning}}
{{/html}}
```

will display

{{warning}}This is a warning message.{{/warning}}

The wiki syntax inside the macro is interpreted

```
{{html wiki="true" clean="false"}}
<html>
  <head><strong>List</strong></head>
  <li>* Item1</li>
  <li>* Item2</li>
</html>
{{/html}}
```

will display

List

- * Item1
- * Item2

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro

ID Macro

The ID macro is similar to the HTML anchor element and it allows you to define a reference or a location in a page.

Usage

{{id name="<name>" /}}

Parameters

Name	Optional	Allowed values	Description	Default value
name	No	Any string	The value of the identifier	None

Example

In the example below, the link will redirect to the last paragraph (i.e. the one starting with "By default, results are sorted...").

= Solr Search =
The new XWiki Enterprise default search engine is Solr, based on Apache Solr which has its own index, separate from the Lucene index. You may access the Solr search page by navigating to Main.SolrSearch.
[[Click here to jump to the last paragraph of this section>>||anchor="lastParagraph"]]
The Solr search module automatically indexes the changes made on the wiki and at each startup a background job checks if the Solr index and the database are synchronized. If not, the job only applies the updates.
The manual indexing is still available because it is more stable so, in case you need to perform it, follow the steps described in the "Search Configuration" documentation page.
{{id name="lastParagraph" /}}
By default, results are sorted by relevance i.e. depending on the scoring of each document. You may also choose to filter the results by name, date or the last author. If the keyword is not found in the document title, the wiki name or the space name but in content of the wiki document, the search results will also display excerpts in which each occurrence will be highlighted.

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro

- Container Macro
- Comment Macro
- Code Macro
- Chart Macro
- Cache Macro
- Box Macro

Include Macro

The Include macro allows you to include other pages in the current page.

Usage

```
{{include reference="<reference>" type="<document>" section="H<section>" context="<new|current>"/}}
{{include document="<document>" type="<document>" section="H<section>" context="<new|current>"/}}
```

Parameters

Name	Optional	Allowed Values	Default Value	Description
reference	No	an entity reference	*none*	Indicates the name of the entity to include.
type	Yes	*document*	*document*	Indicates the type of the entity but for now only *document* is supported.
section	Yes	*string*	*none*	Indicates the anchor to the section to include in the target document (E.g.{{include reference="SpaceName.DocumentToInclude" section="HSectionName"/}}).
document	No	a wiki document	*none*	Represents the name of the document to include. Note that this parameter is deprecated and that you should use the `reference` parameter instead.
context	Yes	*new/current*	*current*	Defines whether the included page is executed in its separated execution context or in the context of the current page. If "context=new" you should use the `display` macro instead.

Examples

```
{{include reference="xwiki:Sandbox.WebHome" type="document" context="new" /}}
{{include reference="xwiki:Sandbox.WebHome" type="document" section="HStyles" context="new" /}}
```

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview

Info Message Macro

The Information Message macro displays an information message in a styled box format.

Usage

```
{{info}}Your custom message.{{/info}}
```

To learn how to further customize the box format, go to the Notification Widget Tutorial.

Example

```
{{info}}
The wiki will be upgraded today between 9AM and 2PM and it will be in "read-only" mode.
{{/info}}
```

will display

The wiki will be upgraded today between 9AM and 2PM and it will be in "read-only" mode.

Related Pages

- **User Guide**
  - XWiki Dashboard Application
- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Office Macro

The Office macro displays an Office attachment as HTML inside the current document.

In order for the Office macro to work, you first need to make sure you have an Open Office/Libre Office server installed and running.

Usage

{{office attachment="<fileName>" filterStyles="<true|false>" /}}

Parameters

Name	Optional	Allowed values	Description	Default value
attachment	No	A document reference	The name of the attachment to display	None
filterStyles	Yes	true false	A flag stating whether to filter the styles from the parsed content	true

Example

{{office attachment="xwiki:Sandbox.Test@DocumentationPresentation2013.odp" filterStyles="false"/}}

will display

Related Pages

- **User Guide**
  - WYSIWYG Editor
  - The Office Importer Application
  - Page Export Formats
  - Page Attachments

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro
- **Admin Guide**
  - Configure the Office Server

Put Footnotes Macro

The Put Footnotes macro is used with the Footnote macro for displaying the footnotes declared in the document content at the current position. In case the document does not contain footnotes, nothing is displayed. Also, if the explicit call of the macro is missing, the macro is called by default at the end of the document.

Usage

{{putFootnotes /}}

Example

In order to create a link, you usually select the text that you would like to become the label of the link and then choose the right option from the Link menu based on the type of link you wish to create. You can change the label of the link later, in the link creation process, by editing the link after it was created or directly in the editing area.
{{footnote}}
You cannot create links inside links and you cannot create a link if the selected label spans through multiple blocks of text (paragraphs, headings, lists etc.). This last constraint is due to the fact that a link has to be in-line.
{{/footnote}}
You can use an image as the label of a link. To do this, you have to select the image and follow the steps for the type of link you wish to create. Note that you won't be able to edit the link label during the link creation process. If you want to change the label of an image link after it was created, then edit the image.
{{footnote}}
The full format of an image is either **##image~: (reference)##** or **##~[~[image~: (reference) {||parameters}]]##**
{{/footnote}}
{{putFootnotes /}}
##queryString##: Allows queries to be passed to the server when creating the download link for the referenced image. Example: ##queryString="width=100&height=800&keepAspectRatio=true"## (**##keepAspectRatio=true## will fail if the ##width## and ##height## parameters are specified in addition to ##queryString##!**)

will display

In order to create a link, you usually select the text that you would like to become the label of the link and then choose the right option from the Link menu based on the type of link you wish to create. You can change the label of the link later, in the link creation process, by editing the link after it was created or directly in the editing area. Be careful though that by changing the label in the link creation process or by editing the link, the label loses its formatting (see Text Formatting). If you don't want to lose the formatting, then edit the link label directly inside the editing area. Selecting the label before choosing a type of link from the "Link" menu is not required: you will be asked to provide a label during the link creation process and this label will be inserted at the caret position. Moreover, you can select an image as the link label. [1]

You can use an image as the label of a link. To do this, you have to select the image and follow the steps for the type of link you wish to create. Note that you won't be able to edit the link label during the link creation process. If you want to change the label of an image link after it was created, then edit the image. [2]

---

1. ^ You cannot create links inside links and you cannot create a link if the selected label spans through multiple blocks of text (paragraphs, headings, lists etc.). This last constraint is due to the fact that a link has to be in-line.
2. ^ The full format of an image is either `image:(reference)` or `[[image:(reference)||parameters]]`

`queryString`: Allows queries to be passed to the server when creating the download link for the referenced image. Example: `queryString="width=100&height=800&keepAspectRatio=true"` (`keepAspectRatio=true` **will fail if the** `width` **and** `height` **parameters are specified in addition to** `queryString`**!**)

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro

- Code Macro
- Chart Macro
- Cache Macro
- Box Macro

Python Macro

The macro adds the ability to write scripts in Python in wiki pages. This is a shortcut for the Script Macro equivalent to `{{script language="python"}}` to execute python scripts based on any JSR-223 Python provider like Jython.

Usage

{{python}}
print "Hello World"
{{/python}}

Parameters Definition

Name	Optional	Allowed values	Default value	Description
language	yes	the identifier of the JSR 223 engine		Indicates which engine to use to execute the provided script.
output	yes	true/false	true	Indicates the output result has to be inserted back in the document.
wiki	yes	true/false	true	indicates if the result of the script execution has to be parsed by the current wiki parser. If not, it is put into a verbatim block.
jars	yes	comma-separated list of JARs that will be added to the script execution class loader	none	

Example

{{python}}
print "Hello World"
{{/python}}

Result

Hello World

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro

RSS Macro

The RSS macro displays the content of an RSS feed.

Usage

{{rss feed="<feed>" content="<true|false>" count="<count>" css="<true|false>" image="<true|false>" width="<px|%>" decoration="<true|false>" /}}

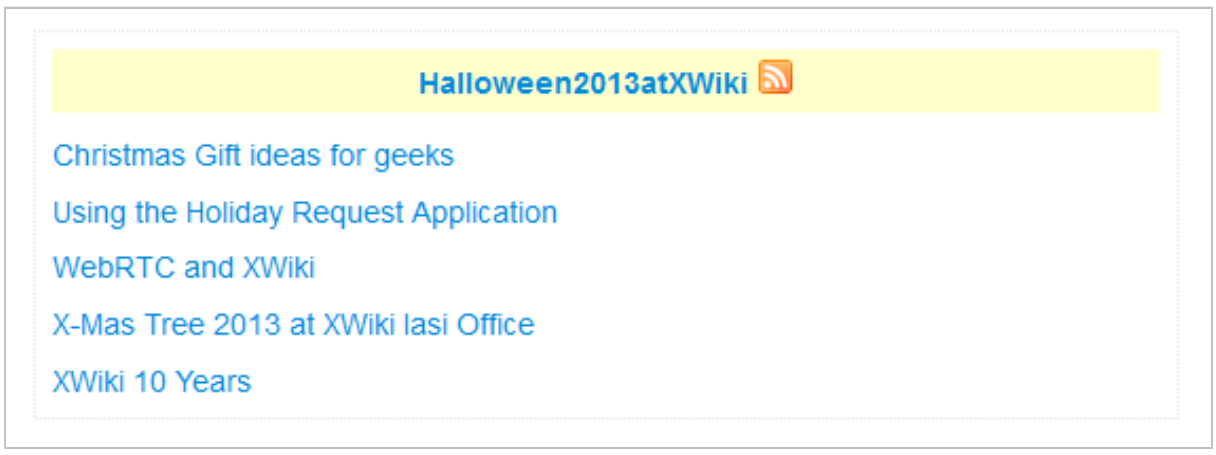
Parameters

Name	Optional	Allowed Values	Default Value	Description
feed	No	An URL	None	The URL of the RSS feed
content	Yes	• true • false	false	A flag stating whether to display the feed content next to the feed item link
count	Yes	An integer	If no value is specified, the macro will display all items	The number of feed items to display
css	Yes	• true • false	false	If the flag is set to "true", the macro adds the dedicated CSS classes from "colibri.css" (`rssfeed`, `rssitem`, `rsschanneltitle`) that can be overridden via the skin page
image	Yes	• true • false	false	A flag stating whether to display the image of the feed
width	Yes	A string	30%	The width of the box containing the RSS output, in px or in %
decoration	Yes	• true • false	true	A flag stating whether to display UI decorations around the feed and feed entries

Examples

{{rss feed="http://xml.feedcat.net/xwikiblogen" content="true" count="5" width="50%" /}}

will display



```
{{rss feed="http://api.flickr.com/services/feeds/photos_public.gne?format=rss_200&tags=xwiki" content="true" count="3"
image="true" width="50%" /}}
```

will display

Recent Uploads tagged xwiki

Ludovic Dubost (XWiki SAS)

xwiki posted a photo:

Ludovic Dubost

xwiki posted a photo:

La salle

xwiki posted a photo:

Related Pages

- **User Guide**
  - XWiki Watchlist Application
  - The Blog Application Overview
  - RSS Feed For The Blog
- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Script Macro

Executes a script, implementing the JSR-223 API.

Usage

```
{{script language="scriptengine"}}
print "some" + " " + "script" + " " + "content"
{{/script}}
```

The **xwiki** and **xcontext** variables are defined by default and represent the XWiki and Context API objects.

Parameters Definition

Name	Optional	Allowed values	Default value	Description
language	yes	the identifier of the JSR 223 engine		Indicates which engine to use to execute the provided script.
output	yes	true/false	true	Indicates the output result has to be inserted back in the document.
wiki	yes	true/false	true	indicates if the result of the script execution has to be parsed by the current wiki parser. If not, it is put into a verbatim block.
jars	yes	comma-separated list of JARs that will be added to the script execution class loader	none	See below

Specifying Extra JARs

If you have programming rights, it is possible to add JARs that will be available when executing the script by using the `jars` parameters. Their format is a comma-separated list of entries:

- a URL: `[[http://server/path/to/some.jar]]`
- a reference to a JAR attachment located on the current page: `[[attach:some.jar]]`
- a reference to a JAR attachment located on another page: `[[attach:wiki:space.page@some.jar]]`
- a reference to all jars located on a given page: `[[attach:wiki:space.page]]`

Bindings

Some bindings are automatically provided:

Name	Class	Description
xwiki	com.xpn.xwiki.api.XWiki	Represents the XWiki object.
xcontext	com.xpn.xwiki.api.Context	Represents the Context of the request.
request	com.xpn.xwiki.web.XWikiRequest	The servlet request. Generally used to get URL parameters.
response	com.xpn.xwiki.web.XWikiResponse	The servlet response.
doc	com.xpn.xwiki.api.Document	The current document.
cdoc	com.xpn.xwiki.api.Document	This variable represent the current default document (not the translated) from which to access the Comments, Objects and Attachments.
tdoc	`com.xpn.xwiki.api.Document`	It represents the translated document matching the requested language

util	com.xpn.xwiki.api.Util	Utility APIs available to scripting environments under the `util` variable.
msg	com.xpn.xwiki.web.XWikiMessageTool	Provides a internationalization service based on key/property values.
syntaxFactory	org.xwiki.rendering.syntax.SyntaxFactory	Allows the creation of a wiki syntax that the user can use to enter wiki content.
officeimporter		Puts a reference to Office Importer in newly created velocity contexts.

Return

You may directly return the result of the macro execution as a single block, as a list of blocks or as XDOM.

This allows you to cover some use cases like:

- when you already have the block (for instance, you got it from a parsing) and you want to insert it into the document.
- when you want to do a safer escaping.

The way of returning a value from a script depends on the language and the engine used. Note that, for now, this is available only for JSR 223 based macros (all macros except the Velocity macro).

Here is an example using Groovy macro:

```
{{groovy}}
return java.util.Arrays.asList(new org.xwiki.rendering.block.WordBlock("Hello"),
org.xwiki.rendering.block.SpaceBlock.SPACE_BLOCK, new org.xwiki.rendering.block.WordBlock("world"));
{{/groovy}}
```

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Success Message Macro

The Success Message macro displays an information message in a styled box format.

Usage

{{success}}Your custom message.{{/success}}

To learn how to further customize the box format, go to the Notification Widget Tutorial.

Example

{{success}}
The user Anna Fox has been successfully invited to the Marketing wiki.
{{/success}}

will display

The user Anna Fox has been successfully invited to the Marketing wiki.

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Table of Contents Macro

The Table of Contents macro generates a table of contents based on XWiki headings. This macro can also be used with the Box macro.

Usage

```
{{toc start="<start>" depth="<depth>" numbered="<true|false>" scope="<local|page>" /}}
```

Parameters

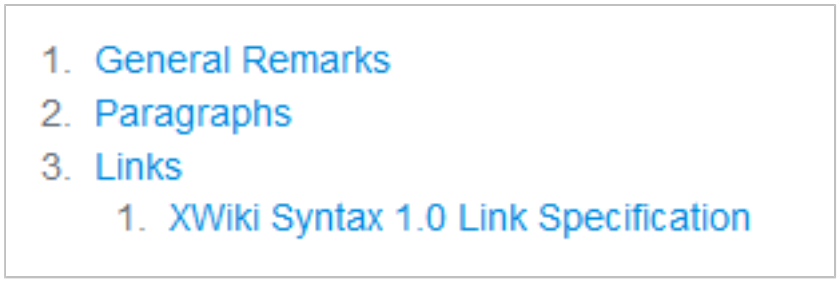
Name	Optional	Allowed values	Description	Default value
start	Yes	1 to 6	The initial level where the table of contents generation should start at. The reason why the second level is the default value is the fact that level 1 is considered as the document title.	2
depth	Yes	1 to 6	The maximum level the table of contents is generated for.	6
numbered	Yes	• true • false	A flag stating whether the macro should generate numbering for titles.	false
scope	Yes	• page • local	If the scope is `local`, only the sections in the current scope will be listed. For instance, in case the macro is called within a section, only its subsections will be listed.	page

Examples

Numbered ToC

```
{{toc start="1" depth="2" numbered="true" scope="page"/}}
= General Remarks =
= Paragraphs =
= Links =
== XWiki Syntax 1.0 Link Specification ==
=== XWiki Syntax 1.0 Image Specification ===
```

will display

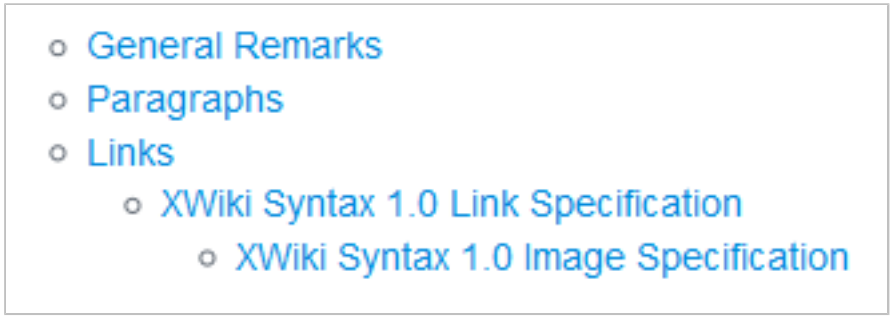


Using the `scope` parameter

```
{{toc start="1" depth="3" scope="page"/}}
= General Remarks =
= Paragraphs =
```

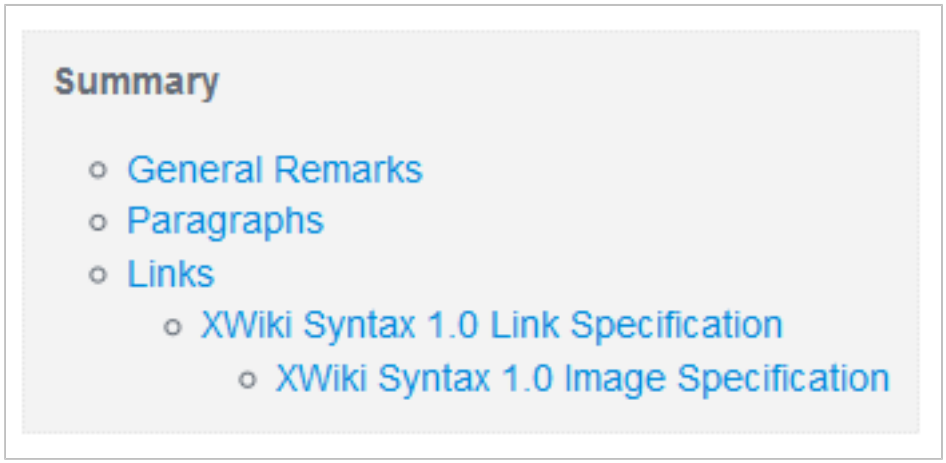
```
= Links =
== XWiki Syntax 1.0 Link Specification ==
=== XWiki Syntax 1.0 Image Specification ===
```

will display



The ToC macro used with the Box macro

```
{{box cssClass="floatinginfobox" title="**Summary**"}}
{{toc/}}
{{/box}}
= General Remarks =
= Paragraphs =
= Links =
== XWiki Syntax 1.0 Link Specification ==
=== XWiki Syntax 1.0 Image Specification ===
```



Using a custom CSS

The custom CSS can be included in a StyleSheetExtension object attached to the page. For the below example the CSS rules for the ".box.floatinginfobox" class are:

```
.box.floatinginfobox ul{
 list-style-type: lower-greek;
 color: #480000;
}
.box.floatinginfobox a:link{
 color: #480000;
}
{{toc /}}
= General Remarks =
= Paragraphs =
= Links =
== XWiki Syntax 1.0 Link Specification ==
=== XWiki Syntax 1.0 Image Specification ===
```

will display

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Translation Macro

The Translation macro allows you to insert a translation message in the page content.

Usage

```
{{translation key="<key>" parameters="<["param0", "param1", "param2"]>" locale="<locale>" /}}
```

The list of the 27 supported locales are available in the documentation page about Internationalization.

Parameters

Name	Optional	Allowed values	Description	Default value
key	No	A string	The key associated to the translation.	None
parameters	Yes	An array	The optional parameter to insert in the translation message.	None
locale	Yes	A valid locale	The language and country in which to translate the key.	None

Example

```
{{translation key="search.panel.header" locale="en_US"/}}
```

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - User Avatar Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro

- Box Macro

User Avatar Macro

The User Avatar macro displays the avatar of a user. If the user has no avatar, the macro displays the standard image "noavatar.png".

Usage

```
{{useravatar username="<username>" width="<width>" height="<height>" /}}
```

Parameters

Name	Optional	Allowed values	Description	Default value
username	No	A string representing the name of an user page.	The user name, the full name or the prefixed full name of an XWiki user.	None
width	Yes	An integer	The width of the avatar image.	The actual width of the image.
height	Yes	An integer	The height of the avatar image.	The actual height of the image.

Example

```
{{useravatar username="xwiki:XWiki.Admin" width="300" /}}
```

will display

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - Velocity Macro
  - Translation Macro
  - Table of Contents Macro

Velocity Macro

The Velocity macro executes a Velocity script and applies current page parser on the result.

Usage

See the example below:

Note that the velocity script is executed before anything else so if you put an xwiki 2.0 **include** inside the **velocity** macro you will not be able to use any velocity macro/variable defined in the included page because the include is done after the velocity is executed. Take care of putting **include** macro outside of **velocity** macro.

Parameters definition

Name	Optional	Allowed values	Default value	Description
output	yes	true/false	true	Indicate if the macro should print something or not.
filter	yes	none/html/indent	indent	Indicate a filter to apply on Velocity content before and after script execution. The default filter (`indent`) removes all first whites spaces of the lines to support code indentation. See Velocity Macro Filter for more details.

Example

```
{{velocity}}
#set($words = ["Some", "velocity", "code"])
#foreach($word in $words)
  $word ##
#end
{{/velocity}}
```

Result

Some velocity code

Velocity Macro Filter

It is possible to filter the Velocity macro content before and after the Velocity Engine execution. You can set the default filter to use in `xwiki.properties` by the setting the property `macro.velocity.filter`. You can also set the filter to use directly when writing the macro by setting the macro `filter` parameter.

Existing Filters

By default the provided filters are:

none

```
macro.velocity.filter=none
macro.velocity.filter=
{{macro filter="none"}}
{{macro filter=""}}
```

Do nothing because this was the behavior before XWiki Platform 1.9.

indent

macro.velocity.filter=indent
{{macro filter="indent"}}

Remove all first white spaces of the lines to support source code indentation without generating white space into the resulting HTML.

```
#if (test)
  Some Text
#end
```

Is the same as:

```
#if (test)
Some Text
#end
```

html

macro.velocity.filter=html
{{macro filter="html"}}

The goal is to make the Velocity script more readable by supporting indentation and code organization with empty lines and putting **if/else/..** on their own lines.

This is done by applying a html to clean the white spaces and new lines. Any group of spaces/new lines is replaced by a space, except for some specific use cases:

- spaces and new lines following a Velocity keyword like **if/else/...** are removed instead of being replaced by a space.
- spaces before a Velocity keyword like **if/else/...** are removed.
- white spaces and new lines before and after **$nl** are removed instead of being replaced by a space.

So for instance:

```
{{velocity}}
The beginning of the line
#if (true)
  #if (true)
    followed by the end of the line.
  #else
    .
  #end
#else
  .
#end
{{/velocity}}
```

will generate:

<p>The beginning of the line followed by the **end** of the line. <p>

instead of:

```
<p>The beginning of the line
<br/>
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;followed by the end of the line.
<br/>
<p>
```

The filter injects the mapping **$sp** and **$nl** in order to be able to force a space of a new line (like putting a **
** in html).

Create a Custom Filter

The Velocity Macro content filters are **xwiki** components. In order to add one, you just need to implement **VelocityMacroFilter** component interface and set the right component attributes.

Related Pages

- **Programming Guide**
  - XWiki Velocity Macros
  - XWiki Scripting
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Warning Message Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

Warning Message Macro

The Warning Message macro displays a warning message in a styled box format.

Usage

{{warning}}Your custom message.{{/warning}}

To learn how to further customize the box format, go to the Notification Widget Tutorial.

Example

{{warning}}
The wiki will be upgraded today between 9AM and 2PM and it will be in "read-only" mode.
{{/warning}}

will display

The wiki will be upgraded today between 9AM and 2PM and it will be in "read-only" mode.

Related Pages

- **Programming Guide**
  - XWiki Rendering Macros in Java
  - XWiki API Reference
  - Velocity Macro
  - User Avatar Macro
  - Translation Macro
  - Table of Contents Macro
  - Success Message Macro
  - Script Macro
  - RSS Macro
  - Python Macro
  - Put Footnotes Macro
  - Programming Overview
  - Office Macro
  - Info Message Macro
  - Include Macro
  - ID Macro
  - HTML Macro
  - Groovy Macro
  - Gallery Macro
  - Formula Macro
  - Footnote Macro
  - Error Macro
  - Display Macro
  - Dashboard Macro
  - Context Macro
  - Content Macro
  - Container Macro
  - Comment Macro
  - Code Macro
  - Chart Macro
  - Cache Macro
  - Box Macro

XWiki Widgets

XWiki provides a collection of front-end components to help you develop in-wiki applications.

Related Pages

- **Programming Guide**
  - XWiki Notification Widget
  - Suggest Widget
  - Programming Overview
  - Modal Popup Widget
  - HTML5 File Upload Widget
  - Confirmation Box Widget
  - Auto-Save Widget

XWiki Notification Widget

The XWiki Notification Widget is a configurable Javascript class used for displaying messages at the bottom of the page (by default).

Parameters

Name	Optional	Allowed values	Description	Default value
text	No	A string	The text displayed by the widget	None
type	Yes	• `plain` • `info` • `warning` • `error` • `inprogress` • `done`	The notification type	plain
options	Yes	• `timeout`: the number of seconds to display the notification. To keep the notification until it will be manually removed, you can set the timeout to `0` or to `false`. • `inactive`: if the option is inactive, the notification won't be displayed after the creation of the object. This is used when you need to manually call `show()` at a later time. • `icon`: a custom image to display. • `background`: the hexadecimal value of a background color. • `onHide`: a custom function that is called when the notification disappears.	Additional configuration parameters for the supported types	See the below paragraph

Configuration Parameters for Supported Types

The source code is available on GitHub:

- notification.js
- notification.css.

- `plain`
  - `timeout`: 5
  - `icon`: none
  - `color`: black
  - `background`: #EEE
- `info`
  - `timeout`: 5
  - `icon`: `(i)`
  - `color`: #28C

        - background: #DDF
    - warning
        - timeout: 5
        - icon: /!\
        - color: 000
        - background: #FFD
    - error
        - timeout: 10
        - icon: (i)
        - color: #900
        - background: #EDD
    - inprogress
        - timeout: false
        - icon: spinning
        - color: #268
        - background: #EEE
    - done
        - timeout: 2
        - icon: (v)
        - color: #090
        - background: #EFD

Example

Supposing you have an application which sends emails to different groups depending on the value of the "sendemail" parameter. When "sendemail=1" you want to display a notification message on the center of the page. A code example would be:

```
{{velocity output="false"}}
#set($discard = $xwiki.ssx.use('Sandbox.InfoNotification'))
#if($request.sendemail == 1 and "${context.action}" == 'view')
  #set($urlview = $doc.getExternalURL('view'))
  #set($creator = $doc.getCreator())
  #set($sender = $xwiki.getDocument(${creator}).getObject('XWiki.XWikiUsers').getProperty('email').value)
  #set($pageattachments=[])#set($user = $context.user)
  #set($emailrecipient = "admin@mydomain.com")
  #set($emailcontent = "
 Hello
<br/><br/>
 An email message was sent to your address <br/><br/>
${urlview}
<br/><br/>")
  #set($ok = $xwiki.mailsender.sendHtmlMessage("$sender", "$emailrecipient", $xwiki.null, $xwiki.null, "An email message
was sent to your address", $emailcontent, $doc.getContent(), $pageattachments))
#end
{{/velocity}}
{{velocity}}
#if($request.sendemail == 1 and "${context.action}" == 'view')
{{html clean="false"}}
 <script type="text/javascript">
  document.observe('xwiki:dom:loaded', function(){
    new XWiki.widgets.Notification('Confirmation message.', 'info', '20');
  });
 </script>
{{/html}}
#end
{{/velocity}}
```

We want to display the message to the center of the page and on a red background and by default the info notification message is displayed at the bottom of the page and on a blue background. We would then need to override the default CSS rules which are available on GitHub. In our case we have overridden the rules for the `.xnotification-container` class in order to display the message on the center of the page:

```
.xnotification-container {
  position: fixed;
  top: 50%;
```

```
  left: 50%;
  margin-top: -100px;
  margin-left: -100px;
  width: 100%;
  text-align: center;
  z-index: 1200;
  display: block !important;
}
```

For the red background, we have overridden the rules for the `.xnotification-info` class as follows:

```
.xnotification-info {
  background-color: #CB2A2A !important;
  color:  #fff;
  border-color:  #fff;
```

Finally, we loading the page with the "?sendemail=1" parameter, the notification message below should appear:

Related Pages

- **Programming Guide**
  - XWiki Widgets
  - Suggest Widget
  - Programming Overview
  - Modal Popup Widget
  - HTML5 File Upload Widget
  - Confirmation Box Widget
  - Auto-Save Widget

Suggest Widget

The Suggest Widget is bundled with XWiki Enterprise and it is triggered when typing something in a text field. The list of suggested results is retrieved from:

- a wiki class
- a custom document
- a REST resource

Source Code

- suggest.js
- suggest.css

Auto-Suggest Using a Custom Document

In order to use an auto-suggest input field, you first need to create a wiki page containing that input field, for instance "Code.SuggestWidgetTest". The HTML code inside the page would be:

```
{{velocity}}
{{html}}
<form>
Input: <input type="text" id="suggestWidget" class="withTip"/>
</form>
{{/html}}
{{/velocity}}
```

The ID of the HTML input is very important because it will be used to bind the text field to the auto-suggest widget. This can be done by using an "XWiki.JavaScriptExtension" object. The "Code" text area will contain the lines below:

```
document.observe("xwiki:dom:loaded", function(){
  var myInput = $('suggestWidget');
 // Create the suggest widget by instantiating XWiki.widgets.Suggest
  var mySuggest = new XWiki.widgets.Suggest(myInput, {
    script: "$xwiki.getURL('Code.SuggestWidgetTestService', 'get', 'outputSyntax=plain')&",
    varname: "q",
    icon: "$xwiki.getSkinFile('icons/silk/page_white_picture.gif')",
    noresults: "No results",
    json: true,
    resultsParameter : "results",
    resultValue : "value",
  });
});
```

where:

- `var myInput = $('suggestWidget');` retrieves the input text element by using its ID. Should the ID change in the HTML, it must also be changed here.
- `script` is the URL end-point that will provide suggestions and it points to data source. In case you want to retrieve field values from an XWiki class or a custom URL to generate the suggested list, you can use the "*/xwiki/templates/suggest.vm*" template.
- `varname` is the name of the request variable on which to pass the text fragment specified by the user. This query parameter can be used by the script to retrieve what has been typed in the text input and use it to compute the results. The query parameter is appended to the script URL, which makes it very important for the script parameter to end with the `&` character in case additional query parameters are specified. The default value is `input`.
- `icon` is used to display an small icon with the passed URL, next to each suggestion.
- `noresults` is the default displayed message when `shownoresults` is enabled and when there are no results to display.
- The `json: true` attribute states whether the output of the data source will be in the JSON format.

- `resultsParameter` and `resultValue` are used to bind the returned JSON structure to the relevant information to be displayed. The returned JSON should be a map that contains a field whose name is the one specified by the `resultsParameter` attribute. This field should contain an array where each element is a map with several fields, including the one specified by the `resultValue` attribute.

For `resultsParameter` the accepted values are:

- `results` for XML results.
- `searchResults` for the REST search.

For `resultValue` the accepted values are:

- `value` for the old suggest.
- `pageFullName` for the REST search.

Other Options for the `XWiki.widgets.Suggest` Constructor

- `cssClass` is the name of the CSS class used by the suggest list. The default value is `ajaxsuggest`.
- `cache` is a flag stating whether to cache the list suggestions returned for a specific input for the lifetime of the current page. The default value is `false`.
- `delay` is the time in milliseconds stating how much to wait after a key press before requesting suggestions from the server. The default value is `500`.
- `hideButton` controls whether a "hide suggestions" button (or 2) is displayed or not. If used, it must be a map with two possible keys:
  - `hideButton.positions` is an array specifying where to place the hide buttons. The accepted values are `top` and `bottom`.
  - `hideButton.text` is the text that should be displayed.
- `highlight` is a flag stating whether the results fragments should be highlighted when matching the typed input. The default value is `true`.
- `method` is the HTTP method for the AJAX request. The default value is `get`.
- `minChars` is the minimum number of characters after which to trigger the suggest. The default value is `1`.
- `offsety` in the value in pixels representing how much to shift the list of suggestions vertically from the normal position. The reason is to allow the user to add decorations between the input and the list. The default value is `0`.
- `parentContainer` is the id of the element that will hold the suggest element which is useful when the enhanced input is not statically positioned. The default value is `body`.
- `resultId` is the name of the JSON parameter or the XML attribute holding the result identifier. The default value is `id`.
- `resultsInfo` is the name of the JSON parameter or the XML attribute holding the result auxiliary information. The accepted values are:
  - `info` for the old suggest.
  - `pageFullName` for the REST search.
- `seps`: in case suggestions should be returned for each token instead of the full text in the input, `seps` should be set to a list of characters that should be used for splitting the input into tokens. If you would rather use the whole text instead, this parameter should be left empty (the default value).
- `shownoresults` is a flag stating what to do when no results match the current input: either display a "no results" message when the flag is set to `true` or hide the suggest box in case no suggestions are available. The default value is `true`.
- `sources` is an array of sources from where to fetch suggestions. If the array is not empty, then the suggest will function in multi-source mode. Every entry should be a map and the parameters should be used as keys in every such map instead of keys in the global options. By default the suggest is in single-source mode.
- `timeout` is the time in milliseconds stating for how long to display the list of suggestions. In case the user doesn't select any of the suggestions before the timeout expires, the list will be cleared. The default value is `2500`.

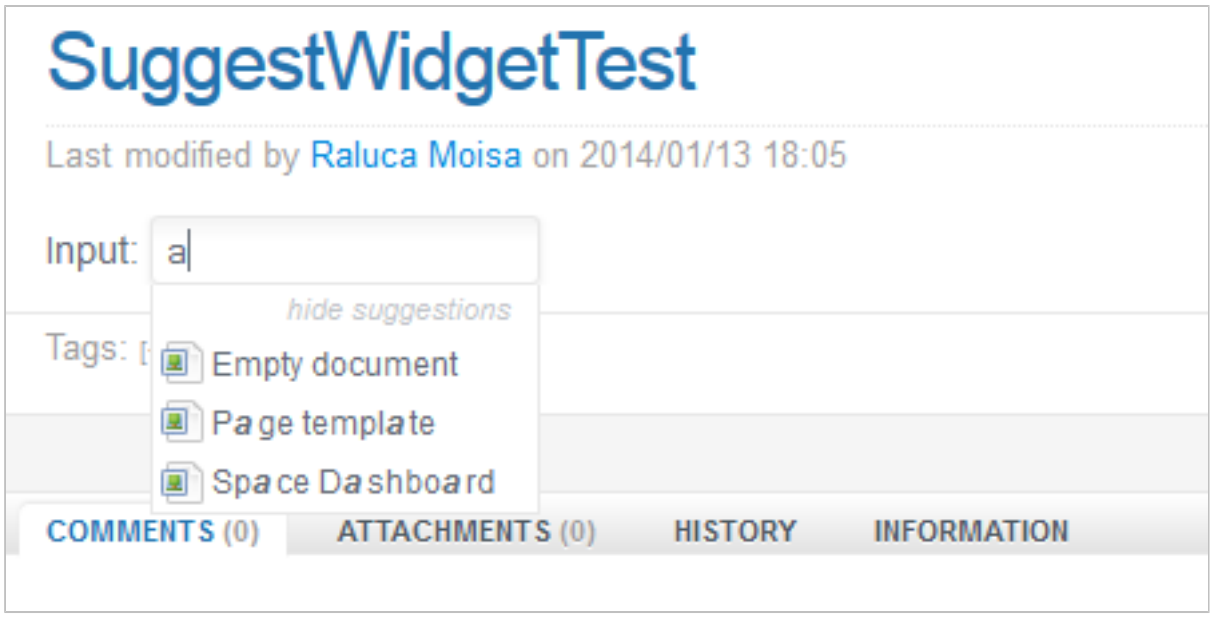
Create a Script Service

Finally, you have to create a script service that should be made available at the URL specified by the `script` attribute of the above JavaScript extension. The service will be available at the *get* address of the page "Code.SuggestWidgetTestService", i.e. */xwiki/bin/get/Code/SuggestWidgetTestService?outputSyntax=plain* which will contain the code below (in "xwiki/2.0" syntax):

```
{{groovy}}
import groovy.json.JsonOutput;
response.setContentType("application/json");
m = [ "results" :
 [
  [ "value" : "Empty document" ],
  [ "value" : "Page template" ],
  [ "value" : "Space Dashboard" ],
```

```
 ]
]
print JsonOutput.toJson(m);
{{/groovy}}
```

If you then refresh the page "Code.SuggestWidgetTest" and type something in the input, you should see the following screen:



Auto-Suggest from the XWiki List of Users and Groups

In order to have an auto-suggest from the list of users and groups (both local and global) you can use the *xwiki/templates/uorgsuggest.vm* template. First, you need to create a wiki page containing that input field as explained in the above section:

```
{{velocity}}
{{html}}
<form>
Input: <input type="text" id="suggestWidgetUsersGroups" class="withTip"/>
</form>
{{/html}}
{{/velocity}}
```

The next step is to bind the input field to the auto-suggest wigdet by using an "XWiki.JavaScriptExtension" object. The "Code" text-area will contain the code below:

```
(function(){
 document.observe('dom:loaded', function () {
  if($('suggestWidgetUsersGroups')) {
    Event.observe($('suggestWidgetUsersGroups'), "focus", function() {
     new XWiki.widgets.Suggest(this, {
          script: '$xwiki.getURL("${doc.fullName}", "view")?
xpage=uorgsuggest&classname=XWiki.XWikiUsers&wiki=global&uorg=user&',
          varname: "input",
          seps: " ,|",
          delay : 200,
          timeout: 5000,
          offsety: 13
      });
    });
  }
 }); // end of doc observe
})();
```

The final result should be

Auto-Suggest Using the `suggest.vm` Template

You can use the "*/xwiki/templates/suggest.vm*" template when you need to have an auto-suggest from an XWiki class (except for the "XWiki.XWikiUsers" class which is presented in the above paragraph) or even from your custom class. Supposing you want to retrieve all the wiki pretty names, the HTML input would be:

```
{{velocity}}
{{html}}
<form>
Input: <input type="text" id="suggestWidgetWikis" name="suggestWidgetWikis" />
</form>
{{/html}}
{{/velocity}}
```

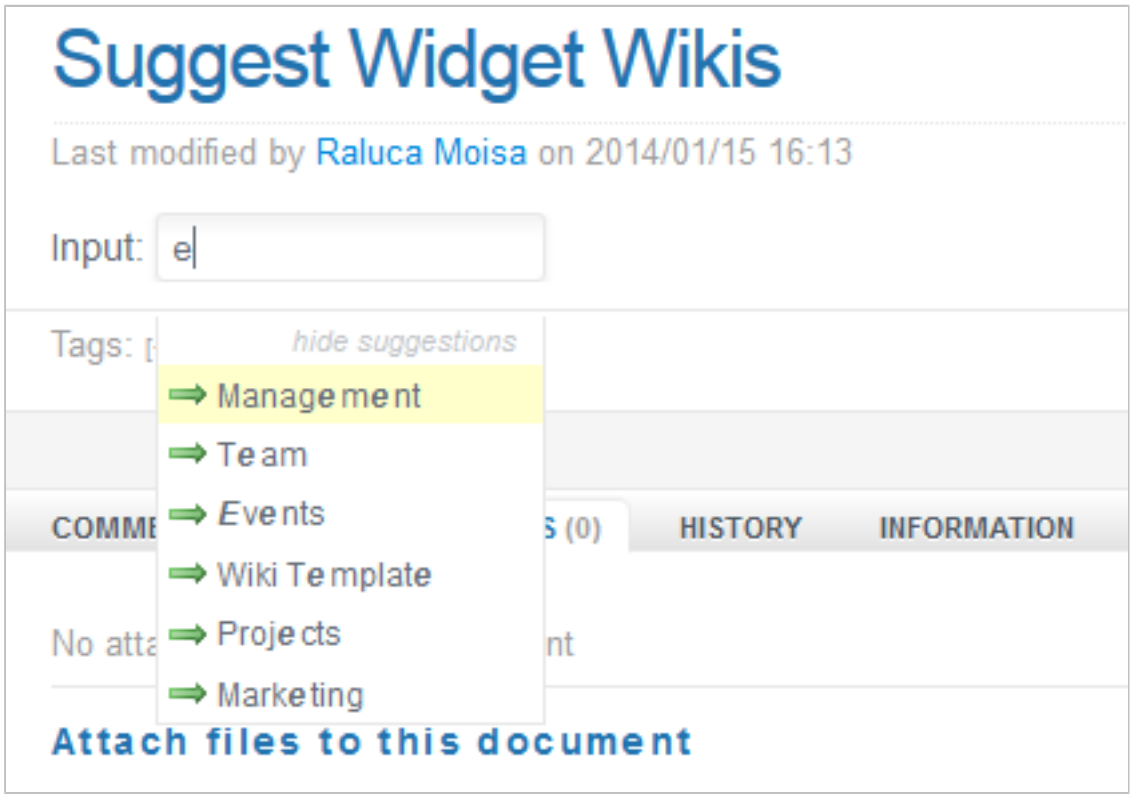
The code for the JavaScript Extension will be

```
(function(){
 document.observe('dom:loaded', function () {
  if($('suggestWidgetWikis')) {
    Event.observe($('suggestWidgetWikis'), "focus", function() {
     new XWiki.widgets.Suggest(this, {
          script: '$xwiki.getURL("${doc.space}.WebHome", "view")?
xpage=suggest&classname=XWiki.XWikiServerClass&fieldname=wikiprettyname&secCol=-&',
          varname: "input",
          seps: " ,|",
          offsety: 13,
          icon: "$xwiki.getSkinFile('icons/silk/arrow_right.gif')"
      });
    });
  }
 }); // end of doc observe
})();
```

where:

- `classname` is the name of the class for the elements of the suggest list ("XWiki.XWikiServerClass" in our case).
- `fieldname` is the name of the class property used for the suggest list.
- `secCol` is the second column of the list of results. For a user defined query, use - for one column and no hidden input. Otherwise the list of results will have two columns and a hidden input.

The result will then be



Auto-Suggest Using a Custom Query

Supposing you need to remove a particular result from the displayed list, like for instance the "Wiki Template", you won't be able to use "*suggest.vm*" anymore. The solution is to create a script service that should be made available at the URL specified by the `script` attribute of the JavaScript extension. For this particular example, the service will be available at the *get* address of the page "Code.SuggestWidgetWikisService":

```
{{velocity}}
$response.setContentType("text/xml")
#set($query = "SELECT prop.value FROM XWikiDocument as doc, BaseObject as obj, StringProperty prop
WHERE doc.fullName=obj.name AND obj.className='XWiki.XWikiServerClass' AND doc.name NOT IN
('XWikiServerWorkspacetemplate', 'XWikiServerClassTemplate') AND prop.id.id=obj.id AND prop.name='wikiprettyname'
ORDER by prop.value desc")
#set($results = $xwiki.search($query, 0, 0))
<?xml version="1.0" encoding="UTF-8"?>
  <results type="8">
   #foreach($res in $results)
    <rs id="$velocityCount" info="">$!{escapetool.xml($res)}</rs>
   #end
  </results>
{{/velocity}}
```

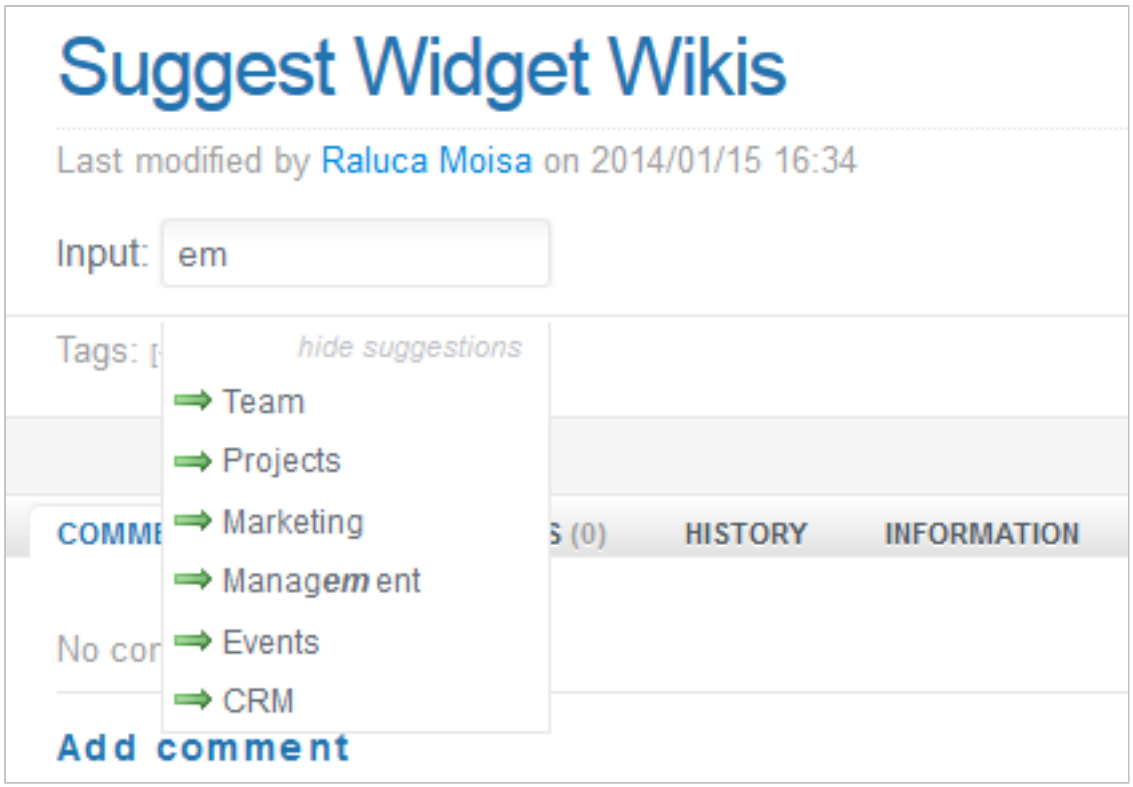
The generated response must be an XML file that has `<results>` as a root node and `<rs>` as children.

The JavaScript extension code will then become:

```
(function(){
 document.observe('dom:loaded', function () {
  if($('suggestWidgetWikis')) {
    Event.observe($('suggestWidgetWikis'), "focus", function() {
     new XWiki.widgets.Suggest(this, {
         script: '$xwiki.getURL('Code.SuggestWidgetWikisService', 'get', 'outputSyntax=plain')&',
         varname: "input",
         seps: " ,|",
         offsety: 13,
         icon: "$xwiki.getSkinFile('icons/silk/arrow_right.gif')"
      });
```

```
    });
  }
 }); // end of doc observe
})();
```

The final result, without "XWikiServerWorkspacetemplate" and "XWikiServerClassTemplate" will be:

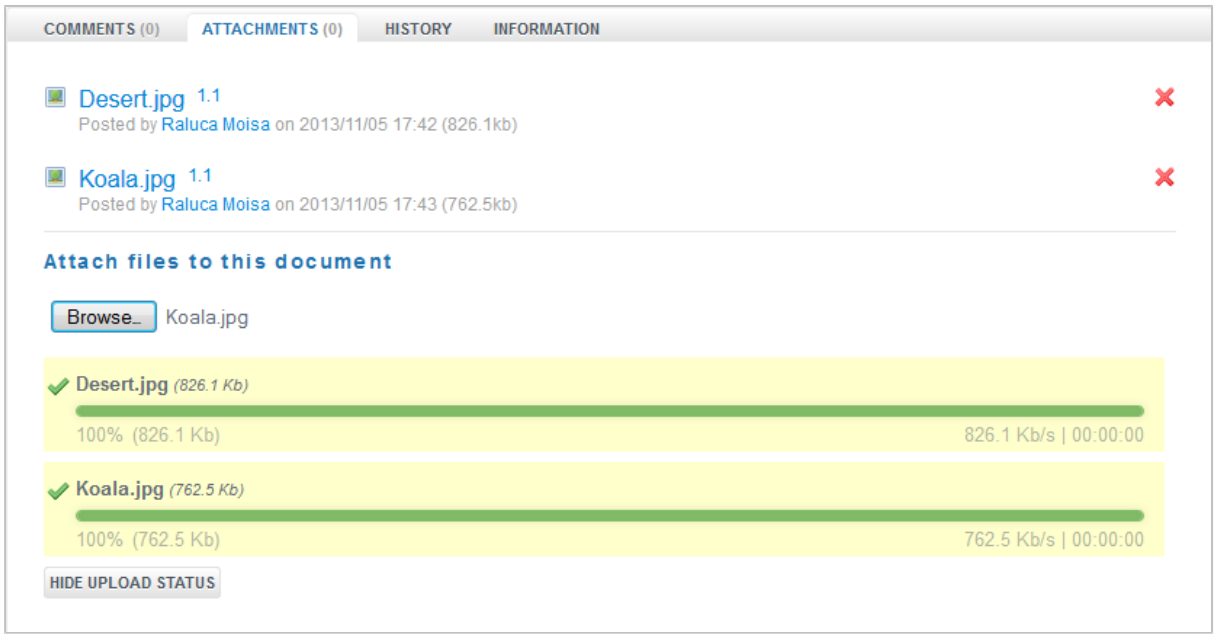


Related Pages

- **Programming Guide**
  - XWiki Widgets
  - XWiki Notification Widget
  - Programming Overview
  - Modal Popup Widget
  - HTML5 File Upload Widget
  - Confirmation Box Widget
  - Auto-Save Widget

HTML5 File Upload Widget

The HTML5 File Upload Widget is bundled with XWiki Enterprise at it provides an interactive upload User Interface in order to enhance the HTML `input` elements of type `file`.



Source Code

- upload.js
- upload.css

Usage

To use the File Upload Widget, you just need to create a new instance of "XWiki.FileUploader" and pass the target `input` element as the first parameter. Additionally, you can pass an optional configuration object as the second parameter.

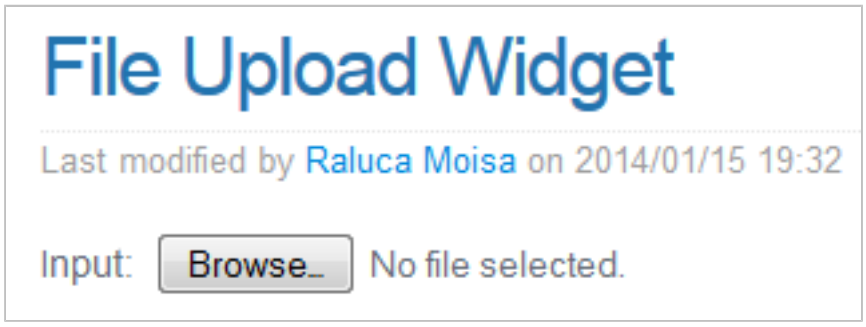
To start, you can create an HTML `input` element of type `file` in a wiki page

```
{{velocity}}
{{html}}
<form>
Input: <input type="file" id="fileUploadWidget" name="fileUploadWidget" />
</form>
{{/html}}
{{/velocity}}
```

then copy the code below in a "XWiki.JavaScriptExtension" object

```
var targetInput = $('fileUploadWidget');
if(targetInput) {
 new XWiki.FileUploader(targetInput, {
   autoUpload: true,
   progressAutohide: true
  });
}
```

which will generate:



Parameters

Option	Description	Accepted value	Default value
maxFilesize	The maximum size of the file.	A number	The maximum attachment size configured for the wiki.
fileFilter	The accepted MIME types.	A valid JavaScript RegExp object.	All MIME types are allowed.
enableFileInfo	A flag stating whether the information (name, file format, size) about each selected file should be displayed.	`true` or `false`	true
enableProgressInfo	A flag stating whether a progress bar should be displayed as each file is uploaded.	`true` or `false`	true
progressAutohide	A flag stating whether the progress information should automatically disappear once all the uploads are completed.	`true` or `false`	false
autoUpload	A flag stating whether the upload should start as soon as the files are selected or whether it should wait for a submit event.	`true` or `false`	true
responseContainer	Indicates the location where the server response is displayed.		If no container is provided, a new `div` input will be created.
responseURL	A custom URL used for retrieving the response after the files are uploaded.	A string	If no URL is provided, the widget will use an existing `xredirect` parameter in the form.

Related Pages

- **User Guide**
  - Page Attachments
- **Programming Guide**
  - XWiki Widgets
  - XWiki Notification Widget
  - Suggest Widget
  - Programming Overview
  - Modal Popup Widget
  - Confirmation Box Widget
  - Auto-Save Widget
- **Admin Guide**
  - Import

Auto-Save Widget

The Auto-Save Widget is bundled with XWiki Enterprise and it is used to enable the periodical automatic saving of a document. At the moment, the widget is available by default only in "Wiki" mode.

To enable the automatic save, just select the "Autosave" checkbox next to the action buttons. Additionally, you can change the interval (in minutes) by specifying an integer value in the dedicated input.

Source Code

- autosave.js
- autosave.css

Usage

To use the Auto-Save widget, start by creating a wiki page and include the following JavaScript resource files:

- `js/xwiki/editors/autosave.js`
- `js/xwiki/actionbuttons/actionButtons.js` - this script file is necessary because the *autosave.js* depends on it. Even though *actionbuttons.js* is already pulled when generating the action buttons, the call is made after the wiki page content has been executed, so you have to explicitly include it.

You can additionally include the `js/xwiki/editors/autosave.css` file, for the case you need to display the widget UI.

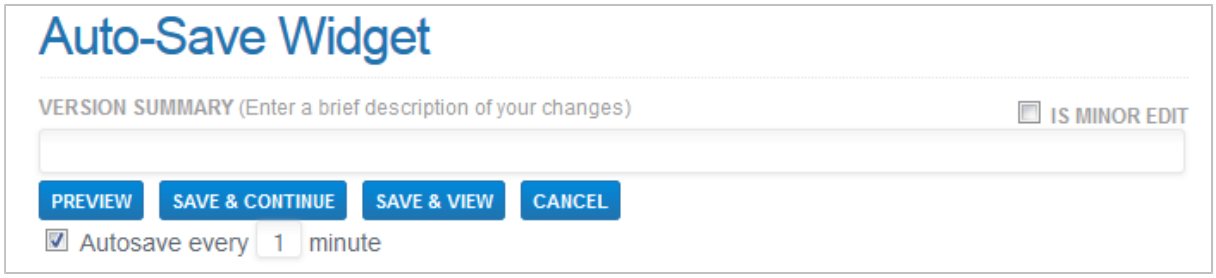
```
{{velocity}}
$xwiki.jsfx.use('js/xwiki/actionbuttons/actionButtons.js', true)##
$xwiki.jsfx.use('js/xwiki/editors/autosave.js')##
$xwiki.ssfx.use('js/xwiki/editors/autosave.css')##
{{/velocity}}
```

Next, add an "XWiki.JavaScriptExtension" object with the following code

```
document.observe('xwiki:dom:loaded', function() {
 new XWiki.editors.AutoSave({
   form : 'inline',
   enabled: true,
   frequency: 1,
   showConfigurationUI: true
  });
});
```

which indicates that the auto-save UI will be displayed in "Inline" mode and the page will be automatically saved every minute.

Finally, edit the page in "Inline" mode and you should see the screen below:



Parameters of the `XWiki.editors.AutoSave` Constructor

Parameter	Description	Default value
form	The ID or DOM node to save.	`xwikieditcontent` which is the main form in wiki edit mode.
enabled	A flag describing the initial state of the auto-save. If `false`, the user has to click the "Autosave" check-box in order to activate it.	false
frequency	The interval in minutes between consecutive saves.	5
showConfigurationUI	A flag stating whether the auto-save UI is displayed or not.	true
disabledOpacity	A number between `0` and `1` which sets the opacity of the frequency input when the auto-save is disabled.	0.2

Related Pages

- **Programming Guide**
  - XWiki Widgets
  - XWiki Notification Widget
  - Suggest Widget
  - Programming Overview
  - Modal Popup Widget
  - HTML5 File Upload Widget
  - Confirmation Box Widget

Confirmation Box Widget

The Confirmation Box Widget is bundled with XWiki Enterprise and it displays a stylized pop-up box asking the user to confirm a given action.

Source Code

- confirmationBox.js
- confirmationBox.css

As you can see in the source code, the Confirmation Box widget is built on top of the Modal Popup class.

Usage

new XWiki.widgets.ConfirmationBox(behavior, interactionParameters);

`XWiki.widgets.ConfirmationBox` Constructor Parameters

behavior

`behavior` is an optional object that defines the "confirm" and "cancel" handlers. When the user confirms an action, the existing `onYes` method is triggered. Otherwise, the existing `onNo` method is triggered. In case the `behavior` parameter is empty, nothing will happen upon confirmation and cancel.

interactionParameters

`interactionParameters` is an object that defines the text elements displayed within the confirmation box:

- a message about the action to confirm; the default value for English is "Are you sure?"
- the text displayed by the "Yes" button
- the text displayed by the "No" button

Example

Create a wiki page and add a link to it that will trigger the deletion of a space:

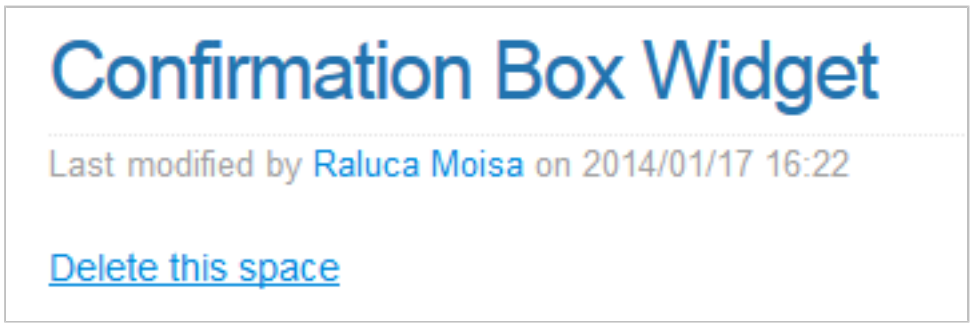
```
{{velocity}}
{{html}}
<a href="#" id="deleteLink">Delete this space</a>
{{/html}}
{{/velocity}}
```

Next, add an "XWiki.JavaScriptExtension" object with the following code:

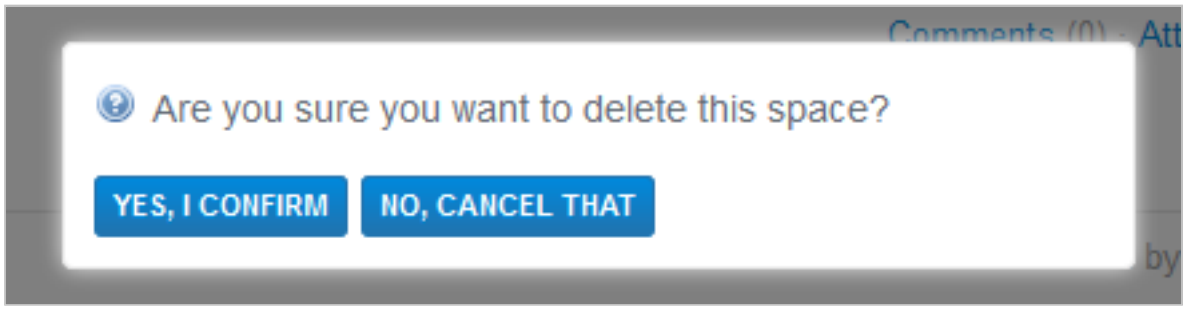
```
// listen to xwiki:dom:loaded to make sure we have XWiki.widgets.ConfirmationBox
document.observe('xwiki:dom:loaded', function() {
 // add onclick listener on element with id 'deleteLink'
  $('deleteLink').observe('click', function(){
   // this is called when click on element with id deleteLink happens
   // init behavior
    var behavior = {
     onYes: function() {
       alert("You have successfully deleted the space.");
     }
    };
   // init interactionParameters
    var interactionParameters = {
     confirmationText: "Are you sure you want to delete this space?",
     yesButtonText: "Yes, I confirm",
     noButtonText: "No, cancel that"
    };
   // instantiate XWiki.widgets.ConfirmationBox
    new XWiki.widgets.ConfirmationBox(behavior, interactionParameters);
```

```
  });
});
```

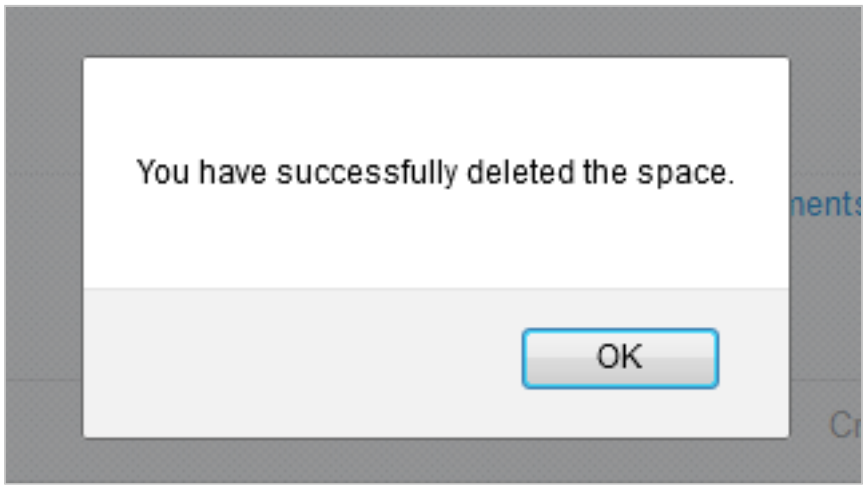
The result should be:



If you click the link, a pop-up with the confirmation message and the button text defined by `interactionParameters` will appear:



In case you decide to confirm the action, you will see a new pop-up with the message defined by the `behavior` parameter.



Related Pages

- **Programming Guide**
  - XWiki Widgets
  - XWiki Notification Widget
  - Suggest Widget
  - Programming Overview
  - Modal Popup Widget
  - HTML5 File Upload Widget
  - Auto-Save Widget

Modal Popup Widget

The Modal Popup Widget is bundled with XWiki Enterprise and it is used as base class for other modal widgets like the Confirmation Box widget, the "Jump to Page" widget and the "Confirmed Ajax Request Box". Since it should not used by itself, the Modal Popup won't display a dialog box.

Source Code

- modalPopup.js
- modalPopup.css
- jumpToPage.js
- jumpToPage.css
- confirmedAjaxRequest.js
- confirmedAjaxRequest.css

Constructor Fields for the "ModalPopup" Class

The constructor for the "ModalPopup" class is called with the following parameters:

- `content` is an object that defines the content of the modal dialog.
- `shortcuts` is an object that defines the shortcuts that will pop up the dialog. For the "Jump to Page" widget, the shortcut keys are "CTRL+META" and "CTRL+G".
- `options` is an object that defines the options for the modal dialog:

Option	Default value
`globalDialog`	`true`
`title`	`""`
`displayCloseButton`	`true`
`extraClassName`	`false`
`screenColor`	`""`
`borderColor`	`""`
`titleColor`	`""`
`backgroundColor`	`""`
`screenOpacity`	`0.5`
`verticalPosition`	`center`
`horizontalPosition`	`center`
`removeOnClose`	`false`
`onClose`	`Prototype.emptyFunction`

Example

We will use the Modal Popup widget to load the wiki page from the Auto-Suggest Widget tutorial that uses a custom service page. The only difference is that the code from the "XWiki.JavaScriptExtension" object attached to "Code.SuggestWidgetTest" will be moved inside a function, "suggestWidget()" as follows:

```
var suggestWidget = function(){
  var myInput = $('suggestWidget');
 // Create the suggest widget by instantiating XWiki.widgets.Suggest
  var mySuggest = new XWiki.widgets.Suggest(myInput, {
    script: "$xwiki.getURL('Code.SuggestWidgetTestService', 'get', 'outputSyntax=plain')&",
    varname: "q",
    icon: "$xwiki.getSkinFile('icons/silk/page_white_picture.gif')",
    noresults: "No results",
    json: true,
    resultsParameter : "results",
    resultValue : "value",
  });
};
```

```
document.observe("xwiki:dom:loaded", function(){
  suggestWidget();
});
```

Next, we will create another wiki page, "Code.ModalPopupTest" and add an "XWiki.JavaScriptExtension" object containing the "ModalPopup" JavaScript class.. This class will be used as base class for the widget that loads the content of "Code.SuggestWidgetTest" and the auto-suggest:

```
// Make sure the XWiki 'namespace' and the ModalPopup class exist.
if(typeof(XWiki) == "undefined" || typeof(XWiki.widgets) == "undefined" || typeof(XWiki.widgets.ModalPopup) ==
"undefined") {
if (typeof console != "undefined" && typeof console.warn == "function") {
    console.warn("[MessageBox widget] Required class missing: XWiki.widgets.ModalPopup");
  }
} else {
XWiki.widgets.MyModalPopup = Class.create(XWiki.widgets.ModalPopup, {
/** Default parameters can be added to the custom class. */
  defaultInteractionParameters : {
  },
/** Constructor. Registers the key listener that pops up the dialog. */
  initialize : function($super, interactionParameters) {
  this.interactionParameters = Object.extend(Object.clone(this.defaultInteractionParameters), interactionParameters || {});
  // call constructor from ModalPopup with params content, shortcuts, options
   $super(
    this.createContent(this.interactionParameters),
      {
      "show"  : { method : this.showDialog,  keys : [] },
      "close" : { method : this.closeDialog, keys : ['Esc'] }
      },
      {
        displayCloseButton : true,
        verticalPosition : "top",
        backgroundColor : "#FFF"
      }
    );
  this.showDialog();
  this.setClass("my-modal-popup");
  },
/** Get the content of the modal dialog using ajax */
  createContent : function (data) {
   var content =  new Element('div', {'class': 'modal-popup'});
  // get page content for the pageURL
   new Ajax.Request(data.pageURL,
    {
      method:'get',
      onSuccess: function(transport){
        var response = transport.responseText || "no response text";
        content.insert(response);
        suggestWidget();
      },
      onFailure: function(){ content.insert('Something went wrong...');
    }
    });
  return content;
  }
});
} // if the parent widget is defined
```

The name of the widget, "MyModalPopup" is defined in the line:

```
XWiki.widgets.MyModalPopup = Class.create(XWiki.widgets.ModalPopup, {}
```

A very important detail is making sure to call the "suggestWidget()" function when making the Ajax request. Otherwise, we will only get the HTML of the page "Code.SuggestWidgetTest", which is an empty input and the auto-suggest won't work:

```
new Ajax.Request(data.pageURL,
```

```
{
 method:'get',
 onSuccess: function(transport){
  var response = transport.responseText || "no response text";
  content.insert(response);
  suggestWidget();
 },
 onFailure: function(){ content.insert('Something went wrong...');
}
});
```

Finally, we will edit the page "Code.ModalPopupTest" in "Wiki" mode and add the following lines:

```
{{velocity}}
#set($pageURL="$xwiki.getURL('Code.SuggestWidgetTest','view','xpage=plain')")
#set ($discard = $xwiki.jsx.use("Code.SuggestWidgetTest"))
{{html}}
<a href="#" onclick="new XWiki.widgets.MyModalPopup({pageURL: '$pageURL'});">Click me to open a pop-up</a>
{{/html}}
{{/velocity}}
```

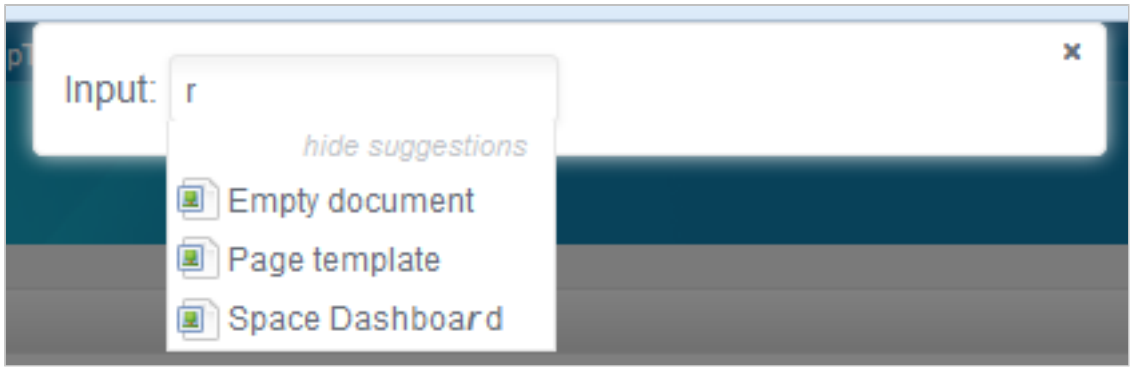
In order for the pop-up to appear, we still need to explicitly load the "XWiki.JavaScriptExtension" object attached to the page "Code.SuggestWidgetTest", even though we have defined the URL for it:

```
#set ($discard = $xwiki.jsx.use("Code.SuggestWidgetTest"))
```

The final result will be:

If we then click on the "Click me to open a pop-up" link, the modal popup window will appear with an HTML input:

Once we start typing something, the auto-suggest will display the possible results that have been defined in the page "Code.SuggestWidgetTestService":



Related Pages

- **Programming Guide**
  - XWiki Widgets
  - XWiki Notification Widget
  - Suggest Widget
  - Programming Overview
  - HTML5 File Upload Widget
  - Confirmation Box Widget
  - Auto-Save Widget

Create and Override a Skin

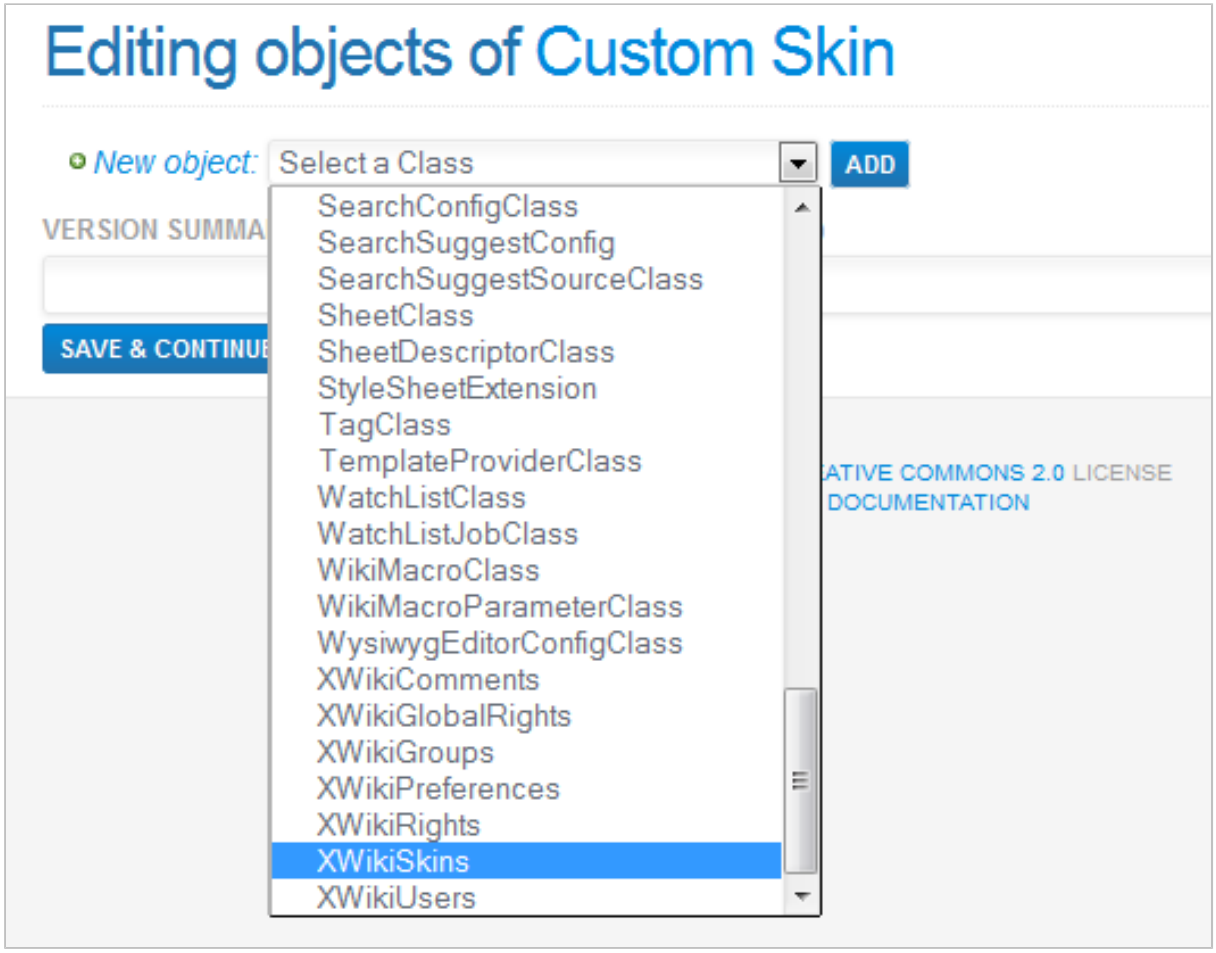
Skin Located on the Filesystem

In this particular case you will have to use the same document as the one used to store the Skin which is located on your filesystem in the folder "/webapps/xwiki/skins/".

Create a Skin Document

To create a new Skin document:

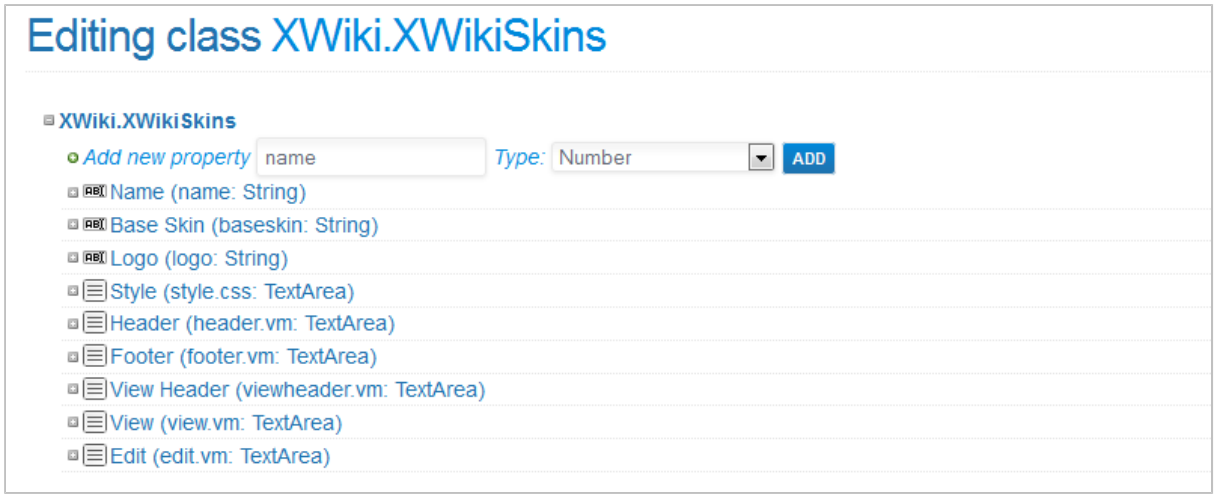
- Create a new page using the "Add" menu in the top left corner.
- Hover the "Edit" menu and click on "Objects".
- From the drop-down list near "New object", choose the "XWiki.XWikiSkins" class, then click on the "Add" button.



- Save the document.

The XWiki.XWikiSkins Class

The base element in overriding a Skin is the "XWiki.XWikiSkins" class (*ServerName/xwiki/bin/edit/XWiki/XWikiSkins?editor=class*). By default, the skin class contains a set of properties allowing you to override some of the templates and files used by the Skin, and also to change the logo displayed in the UI header.



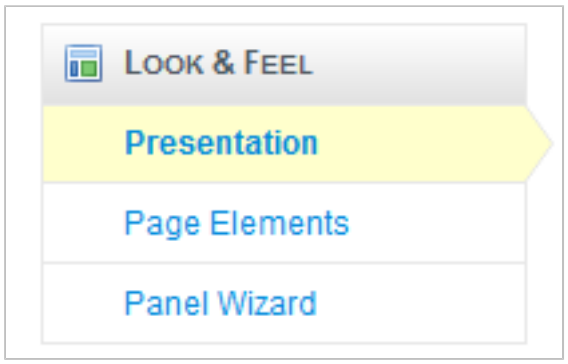
Properties

- "name" represents the pretty name of the skin
- "baseskin" is the base skin overridden by the custom skin; the default value is "colibri"
- "style.css" is the preferred or default stylesheet of the skin
- "header.vm" corresponds to the "/webapps/xwiki/templates/header.vm" template which defines the document headers
- "footer.vm" corresponds to the "/webapps/xwiki/templates/footer.vm" template which defines the document footers
- "viewheader.vm" corresponds to the "/webapps/xwiki/templates/viewheader.vm" template which defines the document headers in view mode: the language bar, the document information bar, the "More Actions" menu etc.
- "view.vm" corresponds to the "/webapps/xwiki/templates/view.vm" template which is the general wrapper for view actions
- "edit.vm" corresponds to the "/webapps/xwiki/templates/edit.vm" template which is the general wrapper for edit actions

To learn how to override the skin templates (like PDF templates) follow this link.

Set the Wiki Default Skin

The skin used by the wiki is defined in the wiki preferences page. To get there, hover the "Wiki" menu and click on "Administer Wiki". Next, click on the "Presentation" link from the vertical menu to the left.



In order to use your custom skin, enter its full name in the "Skin" input then click on the "Save" button.

Another way of doing this is by editing the wiki administration page in "Objects" mode (*ServerName/xwiki/bin/edit/XWiki/XWikiPreferences?editor=object*). Next, enter the full name of the Skin document in the "Skin" property of the "XWiki.XWikiPreferences" object.

If the skin is located on the filesystem, just enter the name of the folder that stores it, for instance `colibrienhanced`.

Create Alternate Stylesheets

The default or preferred stylesheet of the skin is "style.css" and it can be set in the "Stylesheet" field of the "Presentation" section.

You may also use a number of mutually exclusive alternate stylesheets that you will specify in the "Stylesheets" field.

To create an alternate stylesheet, follow these steps:

- Add a new "TextArea" property to the "XWiki.XWikiSkins" class and name it `mynewstylesheet.css`
- Paste your custom CSS code into the `mynewstylesheet.css` property
- Enter the value "mynewstylesheet.css" in the "Stylesheets" field

Override the Preferred Stylesheet

In order to override the default stylesheet, `style.css`, follow these steps:

- Add a new "TextArea" property to the "XWiki.XWikiSkins" class and name it `mynewstylesheet.css`
- Paste your custom CSS code into the `mynewstylesheet.css` property
- Enter the value "mynewstylesheet.css" in the "Stylesheet" field
- Override the `style.css` property of the "XWiki.XWikiSkins" class and add the following code:

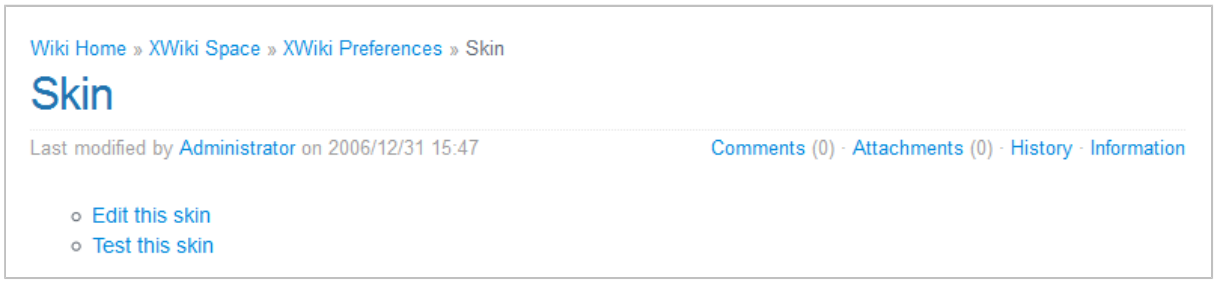
```
@import "mynewfile.css";
```

- Create a file named "mynewfile.css" in your skin folder and add your custom CSS code to it

The important thing is to add a reference to the new component from an existing one.

Customize the Skin Page

To customize the skin page, go to the "Presentation" section of the wiki administration page and click on the "Customize" button next to the "Skin" property. In the skin page, click on "Edit this skin" which will open the document in "Inline" mode.



Override the Skin Resources

Skin resources include the CSS files stored in the skin folder and the .vm files located in the "/webapps/xwiki/templates/" folder. One way of overriding a resource is to extend the "XWiki.XWikiSkins" class by adding a TextArea property with the name of the CSS file or .vm file. Next, edit the wiki default skin in "Objects" mode and copy the default content of the file you wish to override i.e. the content of the file located on your server. Finally, after you are done with the modifications, just click on "Save & View".

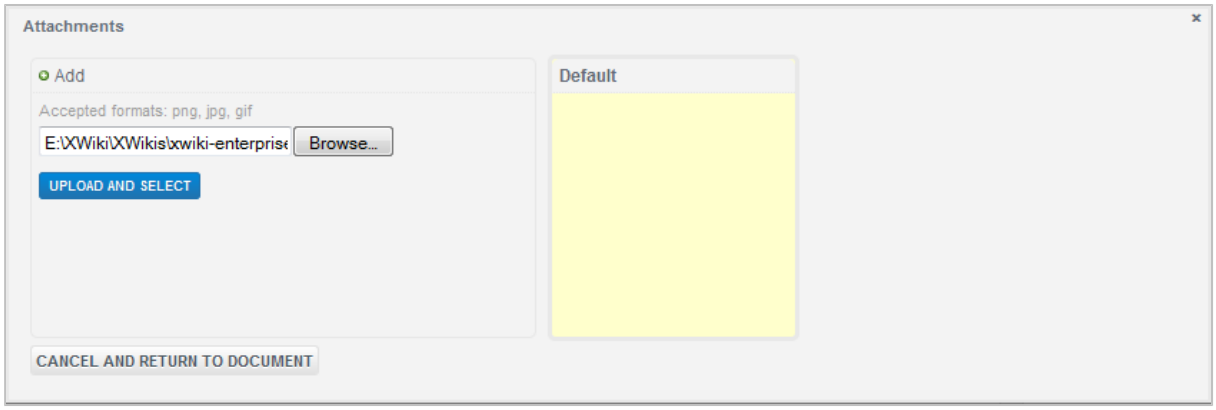
The second way is to attach the modified CSS or the .vm file to the default skin page. This method could seem easier because it doesn't require adding properties to the "XWiki.XWikiSkins" class.

Change the Logo

In order to add `newlogo.jpg` as logo for the current Skin follow these steps:

- Edit the skin page in "Inline" mode
- Click on "Choose an Attachment".

- Select `newlogo.jpg` then click on "Upload and Select".



You will then see the new logo in "Inline" mode.

By default, the displayed logo is the one defined in the color theme page used by the wiki. When the logo is not defined in the color theme, XWiki will display the one attached to the skin file. The reason for this is to allow you to have a different logo for each color theme. For more information, go to the documentation page about how to change the Color Theme logo.

Related Pages

- **User Guide**
  - Editing Modes
- **Programming Guide**
  - XWiki Data Model
  - Skins Extensions
  - Skin Extension Plugin
  - Programming Overview
  - Customize the Look and Feel of the PDF and RTF Export
- **Admin Guide**
  - Color Theme Application

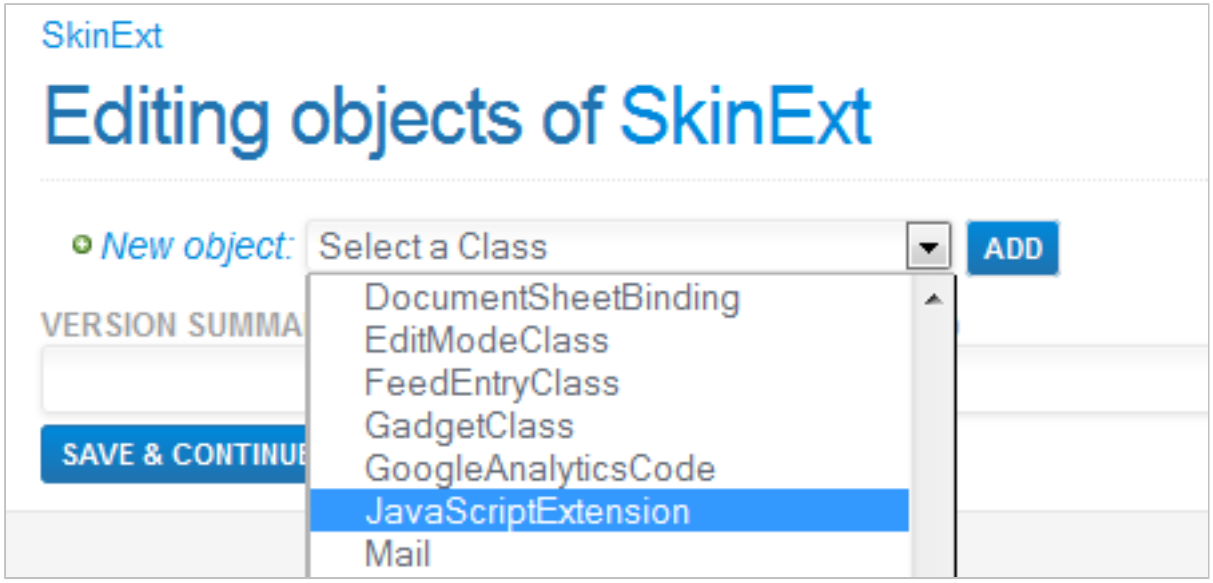
Skins Extensions

The XWiki skins extensions is a mechanism that allows you to customize the layout or just some pages of your wiki, without the need of changing the skin templates or the stylesheets. The "Skin Extension" plugin provides you with the ability to send to the browser extra JavaScript and CSS files that are not part of the actual skin of the wiki. The code for these extensions is defined in wiki objects.

The two supported extensions types are **JavaScript extensions** incarnated by "XWiki.JavaScriptExtension" objects and **StyleSheet extensions** incarnated by "XWiki.StyleSheetExtension" objects.

Create a JavaScript Extension

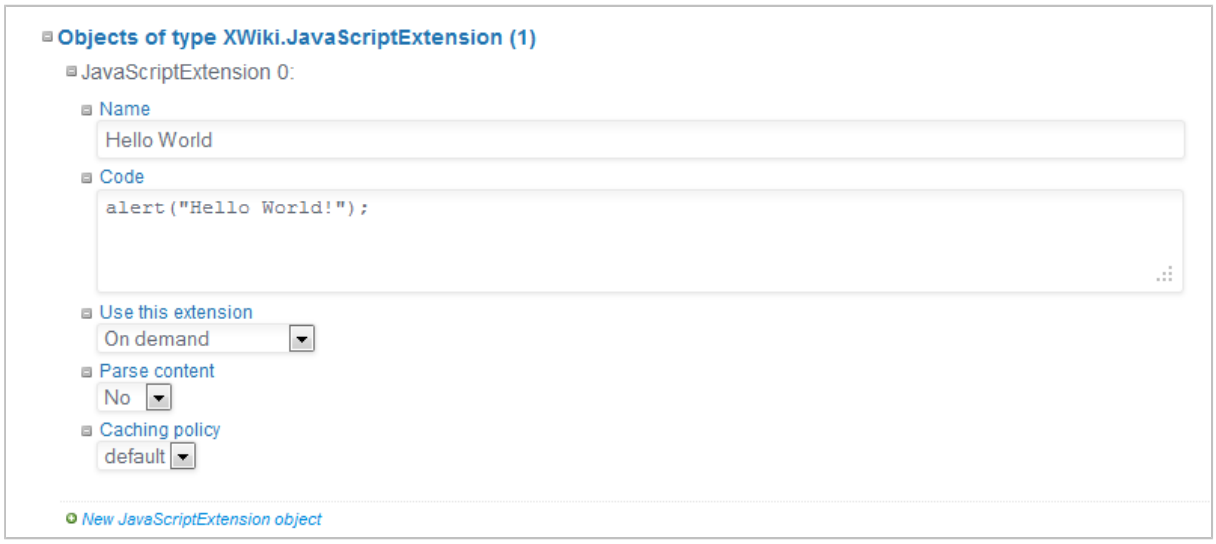
Go to the wiki page you want to create your extension in and edit it in "Objects". Choose the "XWiki.JavaScriptExtension" class from the drop-down list near "New object" and click on the "ADD" button.



The "Use this extension" field has 3 options:

- **Always on this wiki** which indicates that the code will apply on all the pages. In order to execute an extension with this option selected you need to have programming rights.
- **Always on this page**
- **On demand** which will force you to call the extension explicitly to see it executed as shown below

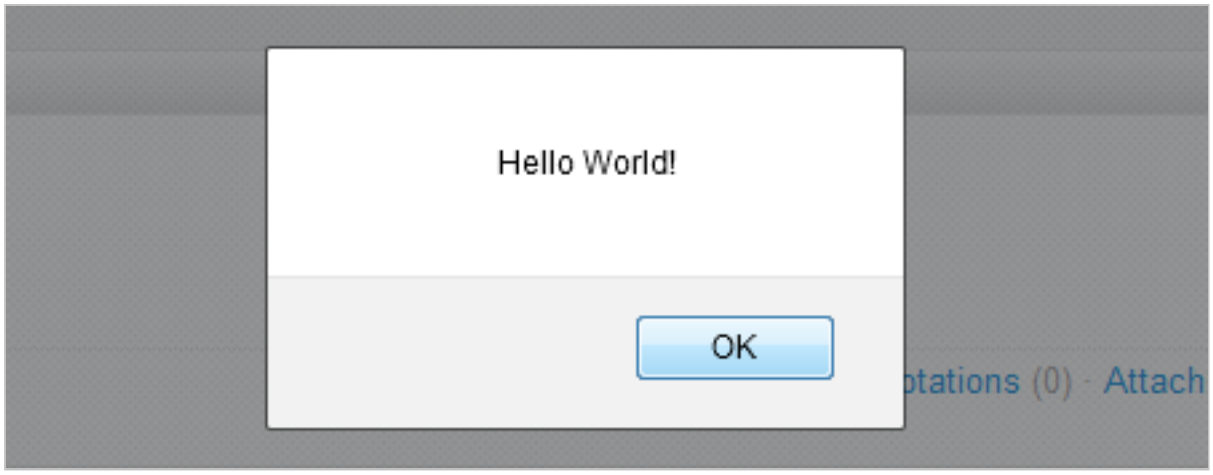
The "Parse content" option set to "No" allows you to write Velocity code. Finally, you can specify a "Caching policy", to tune the HTTP headers that will be returned with the generated JavaScript file.



Given that we have chosen to execute this extension "On demand", we will have to make a call to the Skin Extension plugin by editing the page in wiki mode and adding the following lines:

```
{{velocity}}
$xwiki.jsx.use("$spaceName.$pageName")
{{/velocity}}
```

You should then see an alert box:



Create a StyleSheet Extension

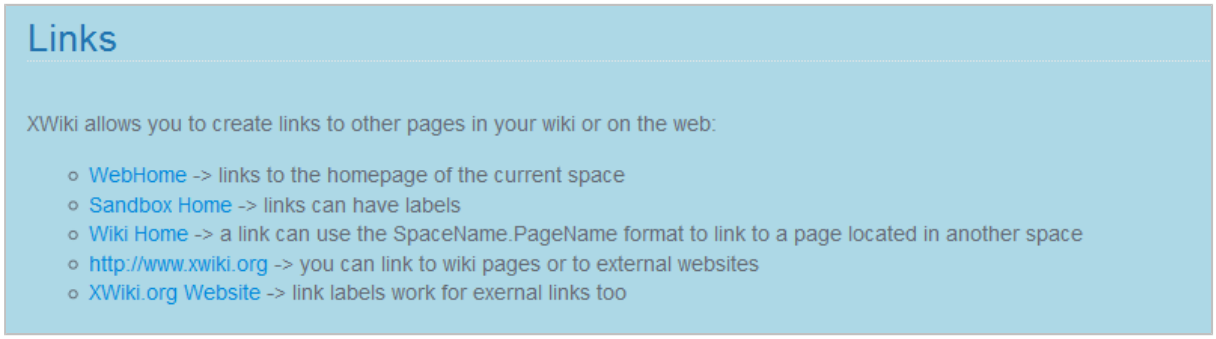
StyleSheet extensions modify the graphical appearance of the wiki pages and they work just like JavaScript extensions, except for the fact that the code is written in CSS.

Create a new page named "XWiki.MyFirstStylesheetExtension", add a "XWiki.StyleSheetExtension" object and enter "My First StyleSheet extension" in the "Name" input. Set the "Use this extension" option to "Always on this page", the "Parse content" option to "No" and select a default cache policy. Next, copy the following lines in the "Code" input:

```
#xwikicontent {
  background-color: lightBlue;
}
```

If you want to use StyleSheet extension on demand, the principle is the same as for JavaScript, except that the name of the plugin is **ssx**, not **jsx**:

```
{{velocity}}
$xwiki.ssx.use("XWiki.MyFirstStylesheetExtension")
{{/velocity}}
```



A document can have as many **ssx** or **jsx** object as it needs, but a skin extension is identified by the name of the document, so in the end an extension is a document. The content of a skin extension is the concatenation of the objects in that document, so it is not possible to write two different extensions in a single document, only different parts of the same extension.

Read about Skins eXtensions with addSizes.js.

Related Pages

- **Programming Guide**
  - Skin Extension Plugin
  - Programming Overview
  - Create and Override a Skin
- **Admin Guide**
  - Extension Manager Application
  - Distribution Wizard
  - Color Theme Application

Configure Keyboard Shortcuts

Customize Keyboard Shortcuts

- Create a new page in your wiki, for example *XWiki.KeyboardShortcuts* and paste the following content inside:
  core.shortcuts.view.edit=e
  core.shortcuts.view.wiki=k
  core.shortcuts.view.wysiwyg=g
  core.shortcuts.view.inline=f
  core.shortcuts.view.rights=r
  core.shortcuts.view.objects=o
  core.shortcuts.view.class=s
  core.shortcuts.view.comments=c
  core.shortcuts.view.attachments=a
  core.shortcuts.view.history=h
  core.shortcuts.view.information=i
  core.shortcuts.view.code=d
  core.shortcuts.view.delete=Delete
  core.shortcuts.view.rename=F2
  core.shortcuts.edit.cancel=Alt+C
  core.shortcuts.edit.backtoedit=Alt+B
  core.shortcuts.edit.preview=Alt+P
  core.shortcuts.edit.saveandcontinue=Alt+Shift+S
  core.shortcuts.edit.saveandview=Alt+S
- Modify the entries in this list at your convenience.
- Save the page.
- Go to the wiki global administration page and click on the "Localization" from the vertical menu to the left.
- Add *XWiki.KeyboardShortcuts* to the "Internationalization Document Bundles" text-box and save the changes.

Add Customized Shortcuts

You can add your own keyboard shortcuts in a few javascript lines:

```
<script type="text/javascript">
//<![CDATA[
 shortcut.add("Ctrl+Alt+n", function() { alert("Hey there! Congratulations on typing this one ;)"); });
//]]>
</script>
{{html clean="false"}}
<script type="text/javascript">
  shortcut.add("Ctrl+Alt+n", function() { alert("Hey there! Congratulations on typing this one ;)"); }, {'propagate':false} );
</script>
{{/html}}
```

Try the snippet by pressing **Ctrl+Alt+n**.

Remove Shortcuts

To remove a shortcut you should adapt this snippet to your needs:

```
<script type="text/javascript">
//<![CDATA[
 shortcut.remove("Ctrl+Alt+n");
//]]>
</script>
{{html clean="false"}}
<script type="text/javascript">
function removeCtrlAltN() {
  shortcut.remove("Ctrl+Alt+n");
```

```
}
</script>
<a onclick="removeCtrlAltN(); return false;" href="#">Click here to try the snippet, it will remove the Ctrl+Alt+n shortcut</
a>.
{{/html}}
```

Remove All Shortcuts at Once

```
<script type="text/javascript">
//<![CDATA[
 for (binding in shortcut.all_shortcuts) {
   shortcut.remove(binding);
  }
//]]>
</script>
{{velocity}}
#if (!$isGuest && $isAdvancedUser)
{{html clean="false"}}
<script type="text/javascript">
//<![CDATA[
function removeAll() {
 for (binding in shortcut.all_shortcuts) {
   shortcut.remove(binding);
  }
}
//]]>
</script>
<a onclick="removeAll(); return false;" href="#">Click here to try the snippet, it will disable all keyboard shortcuts</a>.
{{/html}}
#end
{{/velocity}}
```

Related Pages

- **User Guide**
  - Keyboard Shortcuts

Customize the Look and Feel of the PDF and RTF Export

Customize the PDF Export Look & Feel

You can customize 4 aspects of the PDF export:

- the PDF templates located on your filesystem in the folder *\webapps\xwiki\templates*:
  - `pdf.vm`
  - `pdfhtmlheader.vm`
  - `pdfheader.vm`
  - `pdffooter.vm`
  - `pdftoc.vm`
  - `pdfcover.vm`

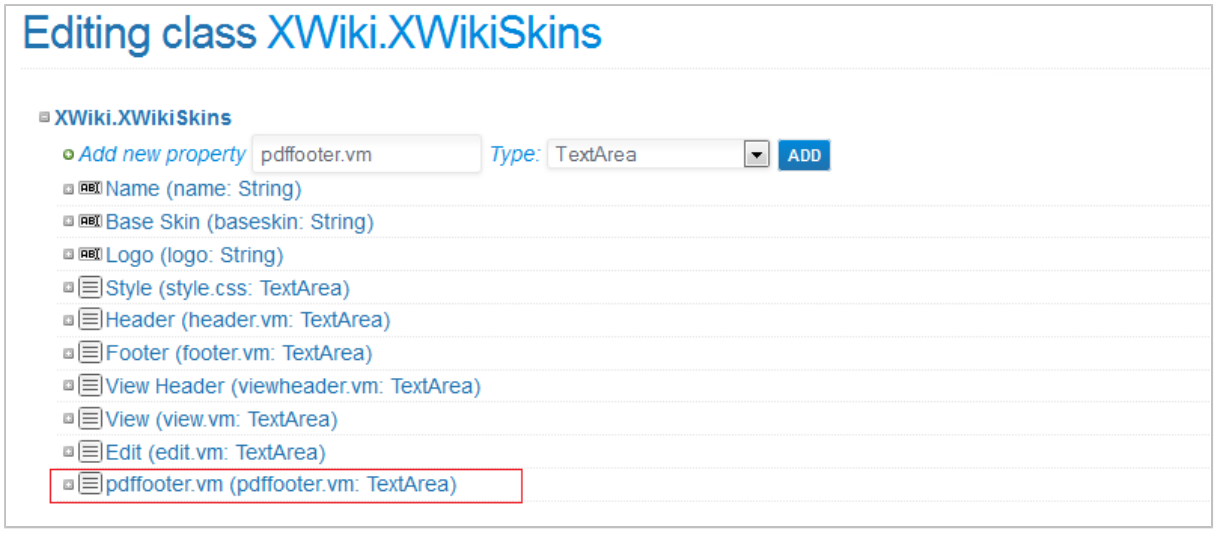
These templates can be overridden in the skin page.

- the CSS rules used to render the content as PDF
- the XHTML2FO XSL transformation
- the FOP XSL transformation

Override the PDF Templates

Customize the PDF Footer

By default, the PDF footer will display the page number, the last author and the date of the last modification. In order to also display a customized message, the template `pdffooter.vm` must be overridden. To do that, edit the Skin class (*ServerName/xwiki/bin/view/XWiki/XWikiSkins?editor=class*) and add a "TextArea" object named `pdffooter.vm`:



Next, edit the skin page (*ServerName/xwiki/bin/edit/XWiki/DefaultSkin?editor=object*) and add the following code to the **pdffooter.vm** property:

```
$msg.Page <span class="page-number"></span> - $msg.get('lastmodifiedby')
$xwiki.getUserName($tdoc.author, false)
$msg.get('lastmodifiedon')
$!xwiki.formatDate($tdoc.date)
<div>
<p>CustomName SAS. All rights reserved. Confidential and proprietary document. Printed Copies are not controlled.</p>
</div>
```

To see the changes, just export any wiki page:

Page 2 - last modified by Silvia Rusu on 2012/02/09 15:13

Customize the PDF Cover

This could be useful when you want for instance to add the company logo to the PDF cover. In order to do this, the template `pdfcover.vm` must be overridden. Just like for the PDF header, a "TextArea" property named `pdfcover.vm` must be added to the "XWiki.XWikiSkins" class (*ServerName/xwiki/bin/view/XWiki/XWikiSkins?editor=class*).

Next, edit the skin page (*ServerName/xwiki/bin/edit/XWiki/DefaultSkin?editor=object*) and add the following code to the **pdfcover.vm** property:

```
<img src="$xwiki.getSkinFile("logo.png")"/>
<div style="text-align: center; width: 100%;">
<h1>
#set($title = "$!pdfdoc.display('title', 'rendered')")
#if($title == '')
  $escapetool.xml($!doc.displayTitle)
#else
  $escapetool.xml($title)
#end
</h1>
<br />
<br />
$!xwiki.getUserName($tdoc.author, false)
<br />
$!xwiki.formatDate($tdoc.date)
</div>
```

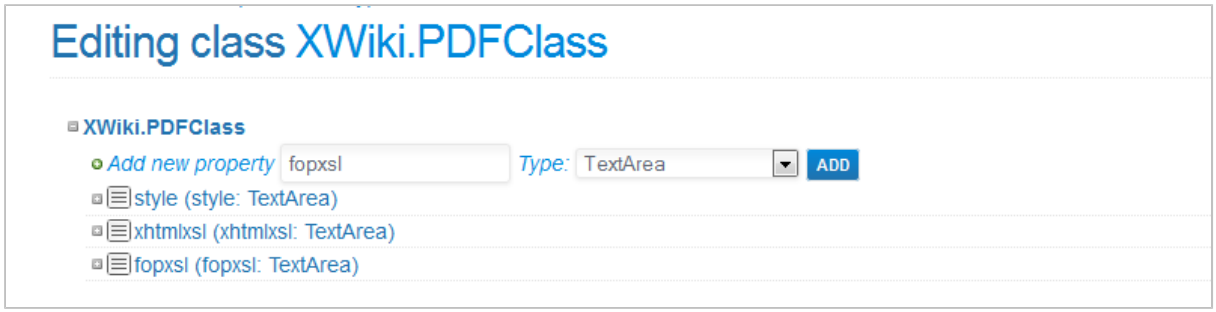
The last step consists of attaching the image "logo.png" to the skin.

Another way of overriding the templates "pdffoofter.vm" and "pdfcover.vm" is to modify them locally, then to attach them to the skin page. This approach does not require adding properties to the class "XWiki.XWikiSkins".

Override the CSS rules

In order to use your own template when exporting a page as PDF, you need to create a class in the "XWiki" space and name it "PDFClass". Next, edit the page in "Class" mode (e.g. *ServerName/xwiki/bin/edit/XWiki/PDFClass?editor=class*) and add the following *TextArea* properties:

- `style` which contains the CSS rules that will override the default `pdf.css` values. By default, there won't be a `pdf.css` file on your filesystem, but you can create it in the folder *\webapps\xwiki\templates* or specify it in your skin page
- `xhtmlxsl` which contains the XHTML2FO XSL transformation that will override the default one
- `fopxsl` which contains the FOP XSL transformation that will override the default one



Next, create the wiki page for which your want to customize the PDF export (e.g. "XWiki.PDFClassTemplate") and add a "XWiki.PDFClass" object to it.

Supposing your wiki page contains a table, you edit it in "Wiki" mode and add a `style` parameter as shown below:

```
(% class="mytable" %)
|=Column 1|=Column 2
| data|data
```

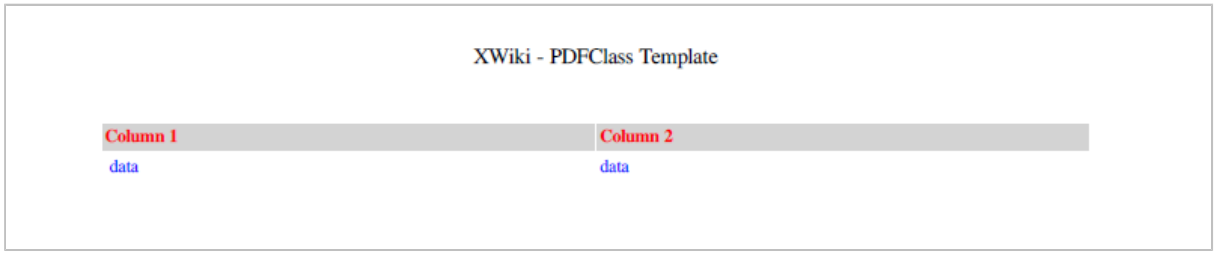
Next, edit the page in "Objects" mode and write your own CSS rules in the "style" property:

In case you would want to use multiple `xhtmlxsl` files, not just the one provided by default, you may attach them to the PDF template page. Next, edit the document in "Objects" mode and add the line below in the "`xhtmlxsl`" field:

$xwiki.getDocument('XWiki.PDFClassTemplate').getAttachment('xhtml2fo.xsl').contentAsString

Because no template is used by default, you need to explicitly specify the `pdftemplate` parameter in the URL in order to use your own template:

ServerName/xwiki/bin/export/XWiki/PDFClassTemplate?format=pdf&pdftemplate=XWiki.PDFClassTemplate.



The available URL parameters for the PDF export are:

- `pages`: you can use it several times, each parameter representing a page to be exported.
- `includechilds`: its numeric value specifies the depth of children pages to include in the PDF export; for instance, if you type `&includechilds=2` in the URL, you will include all the children pages of the page to export (all the pages having this page as parent) and all the children's children.
- `includelinks`: its numeric value specifies the depth of linked pages to include in the PDF export; for example `&includelinks=2` will include all the pages linked from the page to export and the links in those pages.
- `pagebreaks`: if its value is set to `1` then a page break will be inserted between all the exported pages.
- `comments`: if its value is set to `1` then the comments will also be exported.
- `attachments`: if its value is set to `1` then the attached images will also be exported.

Customize the RTF Export Look & Feel

The customization of the RTF export is done just like for the PDF export. The only difference is in the format specified in the URL.

ServerName/xwiki/bin/export/XWiki/RTFTemplate?format=rtf&pdftemplate=XWiki.RTFTemplate

The available URL parameters for the RTF export are the same as for the PDF export.

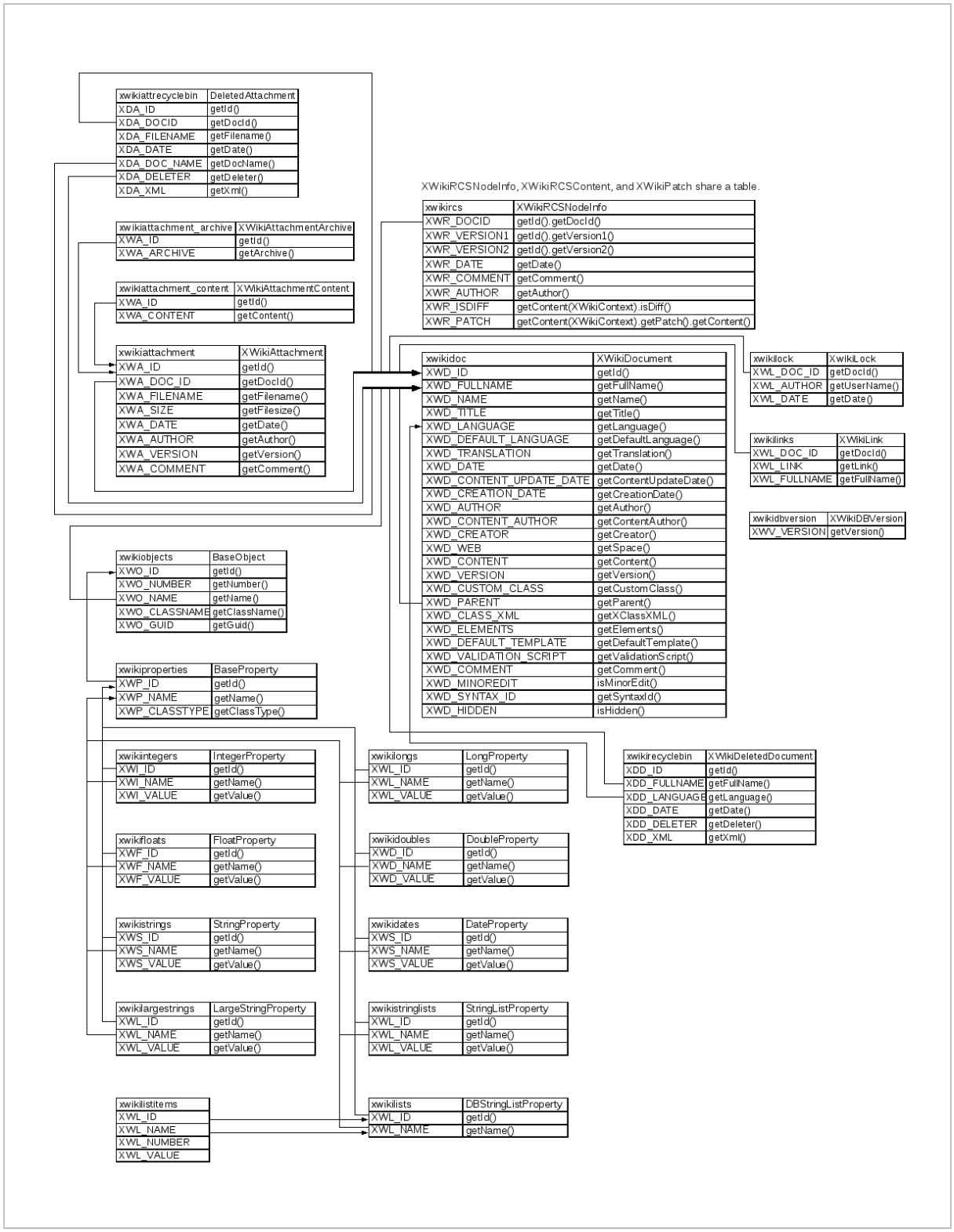
Related Pages

- **User Guide**
  - The Office Importer Application
  - Page Export Formats
  - Editing Modes
- **Programming Guide**
  - XWiki Data Model
  - Create and Override a Skin
- **Admin Guide**
  - Import
  - Export Wiki
  - Configure the Office Server

XWiki Database Schema

Tables of the 'xwiki' Database

Below, there is the a schema diagram of the 'xwiki' database:



Tables

- xwikiattachment
- xwikiattachment_archive
- xwikiattachment_content
- xwikiattrecyclebin
- xwikidates
- xwikidbversion
- xwikidoc
- xwikidoubles
- xwikifloats
- xwikiintegers
- xwikilargestrings
- xwikilinks
- xwikilistitems
- xwikilists
- xwikilock
- xwikilongs
- xwikiobjects
- xwikiproperties
- xwikiircs
- xwikirecyclebin
- xwikistrings

Tables Used for Custom Mapping of XWiki Classes

The custom mappings configuration is done in the `xwiki.hbm.xml` file which is available on GitHub.

- xwikicomments.
- xwikipreferences

For the moment, the "XWikiComments" class is not using the `xwiki.hbm.xml` custom mapping, so the `xwikicomments` table will be empty.

Tables Used by the Statistics Plugin

- xwikistatsdoc
- xwikistatsreferer
- xwikistatsvisit

Tables Used in the Deprecated Table Class Property System

Given that now the class information is stored in the `XWD_CLASS_XML` column of the `xwikidoc` table, the tables below are not used anymore:

- xwikibooleanclasses
- xwikiclasses
- xwikiclassesprop
- xwikidateclasses
- xwikidblistclasses
- xwikinumberclasses
- xwikislistclasses
- xwikistringclasses

Database Indexes

To improve your XWiki instance when running with many documents, you should run the following script to create indexes for your database.

```
  create index xwl_value on xwikilargestrings (xwl_value(50));\\ create index
xwd_parent on xwikidoc (xwd_parent(50));\\ create index xwd_class_xml on
xwikidoc (xwd_class_xml(20));\\ create index xwr_isdiff on xwikircs(xwr_isdiff);
\\ create index xws_number on xwikistatsdoc (XWS_NUMBER);\\ create index
xws_classname on xwikistatsdoc (XWS_CLASSNAME);\\ create index xwr_number on
xwikistatsreferer (XWR_NUMBER);\\ create index xwr_classname on xwikistatsreferer
(XWR_CLASSNAME);\\ create index xwr_referer on xwikistatsreferer (XWR_REFERER(50));
\\ create index xwv_user_agent on xwikistatsvisit (XWV_USER_AGENT(255));\\
create index xwv_cookie on xwikistatsvisit (XWV_COOKIE(255));\\ create index
xwv_classname on xwikistatsvisit (XWV_CLASSNAME);\\ create index xwv_number on
xwikistatsvisit (XWV_NUMBER);\\ create index ase_requestid on activitystream_events
(ase_requestid(200));\\ create index ase_page_date on activitystream_events
(ase_page, ase_date);\\ create index xda_docid1 on xwikiattrecyclebin (xda_docid);\
\
```

Related Pages

- **Programming Guide**

    - XWiki Query Guide
    - The "xwikilock" Table
    - The "xwikidbversion" Table
    - Programming Overview
- **Admin Guide**
    - XWiki Enterprise Setup Overview
    - XWiki Enterprise Installation and Upgrade
    - MySQL Installation and Configuration

The "xwikiattachment" Table

The `xwikiattachment` table corresponds to the "com.xpn.xwiki.doc.XWikiAttachment" class and it stores attachment related information, content and versions excluded.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.doc.XWikiAttachment" table="xwikiattachment">
    <id name="id" type="long" unsaved-value="undefined">
      <column name="XWA_ID" not-null="true" />
      <generator class="assigned" />
    </id>
    <property name="docId" type="long" column="XWA_DOC_ID" index="ATT_DOC_ID" />
    <property name="filename" type="string" column="XWA_FILENAME" length="255" not-null="true" />
    <property name="filesize" type="integer" column="XWA_SIZE" />
    <property name="date" type="timestamp" column="XWA_DATE" index="ATT_DATE" not-null="true" />
    <property name="author" type="string" column="XWA_AUTHOR" length="255" index="ATT_AUTHOR" />
    <property name="version" type="string" column="XWA_VERSION" length="255" not-null="true" />
    <!-- Align the max comment size with the document max comment size (i.e. XWD_COMMENT) -->
    <property name="comment" type="string" column="XWA_COMMENT" length="1023" />
  </class>
```

The columns of the `xwikiattachment` table are:

Column	Data type	"XWikiAttachment" class property	Default value	not-null	Index
XWA_ID (primary key)	bigint(20)	-	0	true	-
XWA_DOC_ID	bigint(20)	docId	null	false	ATT_DOC_ID
XWA_FILENAME	varchar(255)	filename	-	true	-
XWA_SIZE	int(11)	filesize	null	false	-
XWA_DATE	datetime	date	0000-00-00 00:00:00	true	ATT_DATE
XWA_AUTHOR	varchar(255)	author	null	false	ATT_AUTHOR
XWA_VERSION	varchar(255)	version	-	true	-
XWA_COMMENT	varchar(1023)	comment	null	false	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikirecyclebin" Table
  - The "xwikircs" Table
  - The "xwikidoc" Table
  - The "xwikiattrecyclebin" Table
  - The "xwikiattachment_content" Table
  - The "xwikiattachment_archive" Table
  - Attachment Selector Macro

The "xwikiattachment_archive" Table

The `xwikiattachment_archive` table corresponds to the "com.xpn.xwiki.doc.XWikiAttachmentArchive" class and it stores the attachment archive in RCS format.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.doc.XWikiAttachmentArchive" table="xwikiattachment_archive">
    <id name="id" type="long" unsaved-value="undefined">
      <column name="XWA_ID" not-null="true" />
      <generator class="assigned" />
    </id>
    <property name="archive" type="materialized_blob" column="XWA_ARCHIVE" length="1000000000" />
  </class>
```

The columns of the `xwikiattachment_archive` table are:

Column	Data type	"XWikiAttachmentArchive" class property	Default value	not-null	Index
XWA_ID (primary key)	bigint(20)	-	0	true	-
XWA_ARCHIVE	longblob	archive	null	false	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikirecyclebin" Table
  - The "xwikircs" Table
  - The "xwikidoc" Table
  - The "xwikiattrecyclebin" Table
  - The "xwikiattachment_content" Table
  - The "xwikiattachment" Table
  - Attachment Selector Macro

The "xwikiattachment_content" Table

The `xwikiattachment_content` table corresponds to the "com.xpn.xwiki.doc.XWikiAttachmentContent" class and it stores the content of the wiki attachments.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.doc.XWikiAttachmentContent" table="xwikiattachment_content">
    <id name="id" type="long" unsaved-value="undefined">
      <column name="XWA_ID" not-null="true" />
      <generator class="assigned" />
    </id>
    <property name="content" type="materialized_blob" column="XWA_CONTENT" length="1000000000" not-null="true"
/>
  </class>
```

The columns of the `xwikiattachment_content` table are:

Column	Data type	"XWikiAttachmentContent" class property	Default value	not-null	Index
XWA_ID (primary key)	bigint(20)	-	0	true	-
XWA_CONTENT	longblob	content	-	true	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikirecyclebin" Table
  - The "xwikircs" Table
  - The "xwikidoc" Table
  - The "xwikiattrecyclebin" Table
  - The "xwikiattachment_archive" Table
  - The "xwikiattachment" Table
  - Attachment Selector Macro

The "xwikiattrecyclebin" Table

The `xwikiattrecyclebin` table corresponds to the "com.xpn.xwiki.doc.DeletedAttachment" class and it stores information about deleted attachments.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.doc.DeletedAttachment" table="xwikiattrecyclebin" mutable="false">
 <id name="id" column="XDA_ID">
  <generator class="native" />
 </id>
 <natural-id mutable="false">
  <!-- We use docId because most databases do not support indexes on lengthy columns. -->
  <property name="docId" column="XDA_DOCID" type="long" />
  <property name="filename" column="XDA_FILENAME" type="string" length="255" index="XDA_FILENAME" />
  <property name="date" type="timestamp" column="XDA_DATE" index="XDA_DATE" />
 </natural-id>
 <property name="docName" column="XDA_DOC_NAME" type="string" length="255" index="XDA_DOC_NAME" />
 <property name="deleter" type="string" column="XDA_DELETER" index="XDA_DELETER" />
 <property name="xml" type="materialized_clob" column="XDA_XML" not-null="true" length="1000000000" />
</class>
```

The columns of the `xwikiattrecyclebin` table are:

Column	Data type	"DeletedAttachment" class property	Default value	not-null	Index
XDA_ID (primary key and auto incremented)	bigint(20)	-	0	true	-
XDA_DOCID (unique)	bigint(20)	docId	-	true	XDA_DOCID
XDA_FILENAME (unique)	varchar(255)	filename	-	true	XDA_FILENAME
XDA_DATE (unique)	datetime	date	0000-00-00 00:00:00	true	XDA_DATE
XDA_DOC_NAME	varchar(255)	docName	null	false	XDA_DOC_NAME
XDA_DELETER	varchar(255)	deleter	null	false	XDA_DELETER
XDA_XML	longtext	xml	-	true	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikirecyclebin" Table
  - The "xwikircs" Table
  - The "xwikidoc" Table
  - The "xwikiattachment_content" Table
  - The "xwikiattachment_archive" Table
  - The "xwikiattachment" Table
  - Attachment Selector Macro

The "xwikicomments" Table

The `xwikicomments` table stores data from "XWiki.XWikiComments" objects.

For the moment, the "XWikiComments" class is not using the `xwiki.hbm.xml` custom mapping, so the `xwikicomments` table will be empty.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<!-- XWikiComments custom mapping -->
<class entity-name="XWiki.XWikiComments" table="xwikicomments">
 <id name="id" type="long" unsaved-value="undefined">
  <column name="XWC_ID" not-null="true" />
  <generator class="assigned" />
 </id>
 <property name="author" type="string" column="XWC_AUTHOR" length="255" index="COMMENT_AUTHOR" />
 <property name="highlight" type="text" column="XWC_HIGHLIGHT" length="60000" />
 <property name="comment" type="text" column="XWC_COMMENT" length="60000" />
 <property name="replyto" type="integer" column="XWP_REPLYTO" index="COMMENT_REPLYTO" />
 <property name="date" type="timestamp" column="XWP_DATE" index="COMMENT_DATE" />
</class>
```

The columns of the `xwikicomments` table are:

Column	Data type	"XWiki.XWiki Comments" class property	Default value	not-null	Index
XWC_ID (primary key)	bigint(20)	-	0	true	-
XWC_AUTHOR	varchar(255)	author	null	false	COMMENT_AUTHOR
XWC_HIGHLIGHT	longtext	highlight	null	false	-
XWC_COMMENT	longtext	comment	null	false	-
XWP_REPLYTO	int(11)	replyto	null	false	COMMENT_REPLYTO
XWP_DATE	datetime	date	0000-00-00 00:00:00	-	COMMENT_DATE

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **User Guide**
  - Page Comments
  - Annotations Application
- **Programming Guide**
  - The "xwikiproperties" Table
  - The "xwikipreferences" Table
  - The "xwikiobjects" Table
  - Comment Macro
- **Admin Guide**
  - Configure Annotations

The "xwikidates"Table

The `xwikidates` table stores data from XWiki objects of type "Date" and it corresponds to the "com.xpn.xwiki.objects.DateProperty" class which is a sub-class of the "com.xpn.xwiki.objects.BaseProperty" class.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseProperty" table="xwikiproperties">
 <composite-id unsaved-value="undefined">
  <key-property name="id" column="XWP_ID" type="long" />
  <key-property name="name" type="string">
   <column name="XWP_NAME" index="PROP_NAME" />
  </key-property>
 </composite-id>
 <property name="classType" type="string" column="XWP_CLASSTYPE" length="255" />
 <joined-subclass name="com.xpn.xwiki.objects.DateProperty" table="xwikidates">
  <key>
   <column name="XWS_ID" />
   <column name="XWS_NAME" index="XWDATE_NAME" />
  </key>
  <property name="value" type="timestamp" column="XWS_VALUE" index="XWDATE_VALUE" />
 </joined-subclass>
</class>
```

The columns of the `xwikidates` table are:

Column	Data type	"DateProperty" class property	Default value	not-null	Index
XWS_ID (primary key)	bigint(20)	-	0	true	-
XWS_NAME (primary key)	varchar(255)	-	-	true	XWDATE_NAME
XWS_VALUE	datetime	value	0000-00-00 00:00	-	XWDATE_VALUE

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikilongs" Table
  - The "xwikilists" Table
  - The "xwikilistitems" Table
  - The "xwikilargestrings" Table
  - The "xwikiintegers" Table
  - The "xwikifloats" Table
  - The "xwikidoubles" Table

The "xwikidbversion" Table

The `xwikidbversion` table corresponds to the "com.xpn.xwiki.store.migration.XWikiDBVersion" class and it stores information about the database version.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.store.migration.XWikiDBVersion" table="xwikidbversion">
 <id name="version" type="integer" unsaved-value="undefined">
  <column name="XWV_VERSION" not-null="true" />
   <generator class="assigned" />
 </id>
</class>
```

The columns of the `xwikidbversion` table are:

Column	Data type	"XWikiDBVersion" class property	Default value	not-null	Index
XWV_VERSION (primary key)	int(11)	-	0	true	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - XWiki Database Schema
  - The "xwikilock" Table
  - Programming Overview

The "xwikidoc" Table

The `xwikidoc` table corresponds to the "com.xpn.xwiki.doc.XWikiDocument" class and it stores information about the XWiki pages.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.doc.XWikiDocument" table="xwikidoc">
 <id name="id" type="long" unsaved-value="undefined">
  <column name="XWD_ID" not-null="true" />
  <generator class="assigned" />
 </id>
 <property name="fullName" type="string" column="XWD_FULLNAME" length="255" index="DOC_FULLNAME" not-
null="true" />
 <property name="name" type="string" column="XWD_NAME" length="255" index="DOC_NAME" not-null="true" />
 <property name="title" type="string" column="XWD_TITLE" length="255" index="DOC_TITLE" not-null="true" />
 <property name="language" type="string" column="XWD_LANGUAGE" index="DOC_LANGUAGE" length="5" />
 <property name="defaultLanguage" type="string" column="XWD_DEFAULT_LANGUAGE"
index="DOC_DEFAULT_LANGUAGE" length="5" />
 <property name="translation" type="integer" column="XWD_TRANSLATION" not-null="true" />
 <property name="date" type="timestamp" column="XWD_DATE" index="DOC_DATE" not-null="true" />
 <property name="contentUpdateDate" type="timestamp" column="XWD_CONTENT_UPDATE_DATE"
index="DOC_CONTENT_UPDATE_DATE" not-null="true" />
 <property name="creationDate" type="timestamp" column="XWD_CREATION_DATE" index="DOC_CREATION_DATE"
not-null="true" />
 <property name="author" type="string" column="XWD_AUTHOR" length="255" index="DOC_AUTHOR" not-null="true"
/>
 <property name="contentAuthor" type="string" column="XWD_CONTENT_AUTHOR" length="255"
index="DOC_CONTENT_AUTHOR" not-null="true" />
 <property name="creator" type="string" column="XWD_CREATOR" length="255" index="DOC_CREATOR" not-
null="true" />
 <!-- This property is deprecated and should not be used anymore. Use .space instead. -->
 <property name="web" type="string" update="false" insert="false" column="XWD_WEB" length="255" not-null="true" />
 <property name="space" type="string" column="XWD_WEB" length="255" index="DOC_SPACE" not-null="true" />
 <property name="content" type="text" column="XWD_CONTENT" length="200000" not-null="true" />
 <property name="version" type="string" column="XWD_VERSION" length="255" not-null="true" />
 <property name="customClass" type="string" column="XWD_CUSTOM_CLASS" length="255" not-null="true" />
 <property name="parent" type="string" column="XWD_PARENT" length="511" not-null="true" />
 <property name="xWikiClassXML" type="text" column="XWD_CLASS_XML" length="60000" not-null="false" />
 <property name="elements" type="integer" not-null="true" column="XWD_ELEMENTS" />
 <property name="defaultTemplate" type="string" column="XWD_DEFAULT_TEMPLATE" not-null="true" />
 <property name="validationScript" type="string" column="XWD_VALIDATION_SCRIPT" not-null="true" />
 <property name="comment" type="string" column="XWD_COMMENT" length="1023" not-null="true" />
 <property name="minorEdit1" type="boolean" column="XWD_MINOREDIT" index="DOC_MINOREDIT" not-
null="true" />
 <property name="syntaxId" type="string" column="XWD_SYNTAX_ID" length="50" />
 <property name="hidden" type="boolean" column="XWD_HIDDEN" index="DOC_HIDDEN" not-null="true" />
</class>
```

The columns of the `xwikidoc` table are:

Column	Data type	"XWikiDocument" class property	Default value	not-null	Index
XWD_ID (primary key; it is a number generated by taking a hash code of the database name and `XWD_FULLNAME`)	bigint(20)	-	0	true	-
XWD_FULLNAME	varchar(255)	fullName	-	true	DOC_FULLNAME
XWD_NAME	varchar(255)	name	-	true	DOC_NAME
XWD_TITLE	varchar(255)	title	-	true	DOC_TITLE
XWD_LANGUAGE	varchar(5)	language	null	false	DOC_LANGUAGE
XWD_DEFAULT	varchar(5)	defaultLanguage	null	false	DOC_DEFAULT_LANGUAGE

_LANGUAGE					
XWD_TRANSLATION (for regular documents the value is `0` whereas for document translations the value is `1`)	int(11)	translation	0	true	-
XWD_DATE	datetime	date	0000-00-00 00:00:00	true	DOC_DATE
XWD_CONTENT_UPDATE_DATE	datetime	contentUpdateDate	0000-00-00 00:00:00	true	DOC_CONTENT_UPDATE_DATE
XWD_CREATION_DATE	datetime	creationDate	0000-00-00 00:00:00	true	DOC_CREATION_DATE
XWD_AUTHOR	varchar(255)	author	-	true	DOC_AUTHOR
XWD_CONTENT_AUTHOR	varchar(255)	contentAuthor	-	true	DOC_CONTENT_AUTHOR
XWD_CREATOR	varchar(255)	creator	-	true	DOC_CREATOR
XWD_WEB (marked as deprecated and no longer used when updating the table) XWD_WEB	varchar(255) varchar(255)	web space	- -	true true	- DOC_SPACE
XWD_CONTENT	longtext	content	-	true	-
XWD_VERSION	varchar(255)	version	-	true	-
XWD_CUSTOM_CLASS	varchar(255)	customClass	-	true	-
XWD_PARENT	varchar(511)	parent	-	true	-
XWD_CLASS_XML	longtext	xWikiClassXML	null	false	-
XWD_ELEMENTS	int(11)	elements	-	true	-
XWD_DEFAULT_TEMPLATE	varchar(255)	defaultTemplate	-	true	-
XWD_VALIDATION_SCRIPT	varchar(255)	validationScript	-	true	-
XWD_COMMENT	varchar(1023)	comment	-	true	-
XWD_MINOREDIT	bit(1)	minorEdit1	-	true	DOC_MINOREDIT
XWD_SYNTAX_ID	varchar(511)	syntaxId	-	-	-
XWD_HIDDEN	bit(1)	hidden	-	true	DOC_HIDDEN

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikirecyclebin" Table
  - The "xwikilinks" Table
  - The "xwikiattrecyclebin" Table
  - The "xwikiattachment_content" Table
  - The "xwikiattachment_archive" Table
  - The "xwikiattachment" Table
  - Attachment Selector Macro

The "xwikidoubles" Table

The `xwikidoubles` table stores data from XWiki object properties of type "Double" and it corresponds to the "com.xpn.xwiki.objects.DoubleProperty" class which is a sub-class of the "com.xpn.xwiki.objects.BaseProperty" class.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseProperty" table="xwikiproperties">
 <composite-id unsaved-value="undefined">
  <key-property name="id" column="XWP_ID" type="long" />
  <key-property name="name" type="string">
   <column name="XWP_NAME" index="PROP_NAME" />
  </key-property>
 </composite-id>
 <property name="classType" type="string" column="XWP_CLASSTYPE" length="255" />
 <joined-subclass name="com.xpn.xwiki.objects.DoubleProperty" table="xwikidoubles">
  <key>
   <column name="XWD_ID" />
   <column name="XWD_NAME" index="XWDOUBLE_NAME" />
  </key>
  <property name="value" type="double" column="XWD_VALUE" index="XWDOUBLE_VALUE" />
 </joined-subclass>
</class>
```

The columns of the `xwikidoubles` table are:

Column	Data type	"DoubleProperty" class property	Default value	not-null	Index
XWD_ID (primary key)	bigint(20)	-	0	true	-
XWD_NAME (primary key)	varchar(255)	-	-	true	XWDOUBLE_NAME
XWD_VALUE	double	value	null	false	XWDOUBLE_VALUE

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikilongs" Table
  - The "xwikilists" Table
  - The "xwikilistitems" Table
  - The "xwikilargestrings" Table
  - The "xwikiintegers" Table
  - The "xwikifloats" Table
  - The "xwikidates"Table

The "xwikifloats" Table

The `xwikifloats` table stores data from XWiki object properties of type "Float" and it corresponds to the "com.xpn.xwiki.objects.FloatProperty" class which is a sub-class of the "com.xpn.xwiki.objects.BaseProperty" class.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseProperty" table="xwikiproperties">
 <composite-id unsaved-value="undefined">
  <key-property name="id" column="XWP_ID" type="long" />
  <key-property name="name" type="string">
   <column name="XWP_NAME" index="PROP_NAME" />
  </key-property>
 </composite-id>
 <property name="classType" type="string" column="XWP_CLASSTYPE" length="255" />
 <joined-subclass name="com.xpn.xwiki.objects.FloatProperty" table="xwikifloats">
  <key>
   <column name="XWF_ID" />
   <column name="XWF_NAME" index="XWFLOAT_NAME" />
  </key>
  <property name="value" type="float" column="XWF_VALUE" index="XWFLOAT_VALUE" />
 </joined-subclass>
</class>
```

The columns of the `xwikifloats` table are:

Column	Data type	"FloatProperty" class property	Default value	not-null	Index
XWF_ID (primary key)	bigint(20)	-	0	true	-
XWF_NAME (primary key)	varchar(255)	-	-	true	XWFLOAT_NAME
XWF_VALUE	float	value	null	false	XWFLOAT_VALUE

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikilongs" Table
  - The "xwikilists" Table
  - The "xwikilistitems" Table
  - The "xwikilargestrings" Table
  - The "xwikiintegers" Table
  - The "xwikidoubles" Table
  - The "xwikidates"Table

The "xwikiintegers" Table

The `xwikiintegers` table stores data from XWiki object properties of type "Integer" and it corresponds to the "com.xpn.xwiki.objects.IntegerProperty" class which is a sub-class of the "com.xpn.xwiki.objects.BaseProperty" class.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseProperty" table="xwikiproperties">
 <composite-id unsaved-value="undefined">
  <key-property name="id" column="XWP_ID" type="long" />
  <key-property name="name" type="string">
   <column name="XWP_NAME" index="PROP_NAME" />
  </key-property>
 </composite-id>
 <property name="classType" type="string" column="XWP_CLASSTYPE" length="255" />
 <joined-subclass name="com.xpn.xwiki.objects.IntegerProperty" table="xwikiintegers">
  <key>
   <column name="XWI_ID" />
   <column name="XWI_NAME" index="XWINT_NAME" />
  </key>
  <property name="value" type="integer" column="XWI_VALUE" index="XWINT_VALUE" />
 </joined-subclass>
</class>
```

The columns of the `xwikiintegers` table are:

Column	Data type	"IntegerProperty" class property	Default value	not-null	Index
XWI_ID (primary key)	bigint(20)	-	0	true	-
XWI_NAME (primary key)	varchar(255)	-	-	true	XWINT_NAME
XWI_VALUE	int(11)	value	null	false	XWINT_VALUE

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikilongs" Table
  - The "xwikilists" Table
  - The "xwikilistitems" Table
  - The "xwikilargestrings" Table
  - The "xwikifloats" Table
  - The "xwikidoubles" Table
  - The "xwikidates"Table

The "xwikilargestrings" Table

The `xwikilargestrings` table stores data from XWiki objects of type "Text Area" and "Static List". This table corresponds to the "com.xpn.xwiki.objects.StringListProperty" and "com.xpn.xwiki.objects.LargeStringProperty" classes which are a sub-classes of the "com.xpn.xwiki.objects.BaseProperty" class.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseProperty" table="xwikiproperties">
 <composite-id unsaved-value="undefined">
  <key-property name="id" column="XWP_ID" type="long" />
  <key-property name="name" type="string">
   <column name="XWP_NAME" index="PROP_NAME" />
  </key-property>
 </composite-id>
 <property name="classType" type="string" column="XWP_CLASSTYPE" length="255" />
 <joined-subclass name="com.xpn.xwiki.objects.LargeStringProperty" table="xwikilargestrings">
  <key>
   <column name="XWL_ID" />
   <column name="XWL_NAME" index="XWLS_NAME" />
  </key>
  <property name="value" type="text" column="XWL_VALUE" length="60000" />
 </joined-subclass>
 <joined-subclass name="com.xpn.xwiki.objects.StringListProperty" table="xwikilargestrings">
  <key>
   <column name="XWL_ID" />
   <column name="XWL_NAME" />
  </key>
  <property name="textValue" type="text" column="XWL_VALUE" length="60000" />
 </joined-subclass>
</class>
```

The columns of the `xwikilargestrings` table are:

Column	Data type	"LargeString Property" class property	"StringList Property" class property	Default value	not-null	Index
XWL_ID (primary key)	bigint(20)	-	-	0	true	-
XWL_NAME (primary key)	varchar(255)	-	-	-	true	XWLS_NAME
XWL_VALUE	longtext	value	textValue	null	false	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikilongs" Table
  - The "xwikilists" Table
  - The "xwikilistitems" Table
  - The "xwikiintegers" Table
  - The "xwikifloats" Table
  - The "xwikidoubles" Table
  - The "xwikidates"Table

The "xwikilinks" Table

The `xwikilinks` table corresponds to the "com.xpn.xwiki.doc.XWikiLink" class and it stores information about links that can be parsed in each document.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.doc.XWikiLink" table="xwikilinks">
 <composite-id unsaved-value="undefined">
  <key-property name="docId" column="XWL_DOC_ID" type="long" />
  <key-property name="link" type="string" length="255">
   <column name="XWL_LINK" index="XWLNK_LINK" />
  </key-property>
 </composite-id>
 <property name="fullName" type="string" column="XWL_FULLNAME" length="255" />
</class>
```

The columns of the `xwikilinks` table are:

Column	Data type	"XWikiLink" class property	Default value	not-null	Index
XWL_DOC_ID (primary key)	bigint(20)	docId	0	true	-
XWL_LINK (primary key)	varchar(255)	link	-	true	XWLNK_LINK
XWL_FULLNAME	varchar(255)	fullName	null	false	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikirecyclebin" Table
  - The "xwikidoc" Table

The "xwikilistitems" Table

The `xwikilistitems` stores the data of the list items for XWiki objects of type "Database List". The list is stored in the `xwikilists` table.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<joined-subclass name="com.xpn.xwiki.objects.DBStringListProperty" table="xwikilists">
 <key>
  <column name="XWL_ID" />
  <column name="XWL_NAME" index="XWLIST_NAME" />
 </key>
 <list name="list" table="xwikilistitems" lazy="false">
  <key>
   <column name="XWL_ID" />
   <column name="XWL_NAME" index="XWLI_NAME" />
  </key>
  <index column="XWL_NUMBER" />
  <element type="string">
   <column name="XWL_VALUE" index="XWLI_VALUE" />
  </element>
 </list>
</joined-subclass>
```

The columns of the `xwikilistitems` table are:

Column	Data type	Default value	not-null	Index
XWL_ID (primary key)	bigint(20)	0	true	-
XWL_NAME (primary key)	varchar(255)	-	true	XWLI_NAME
XWL_NUMBER (primary key)	int(11)	0	true	-
XWL_VALUE	varchar(255)	null	false	XWLI_VALUE

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikilongs" Table
  - The "xwikilists" Table
  - The "xwikilargestrings" Table
  - The "xwikiintegers" Table
  - The "xwikifloats" Table
  - The "xwikidoubles" Table
  - The "xwikidates"Table

The "xwikilists" Table

The `xwikilists` table corresponds to the "com.xpn.xwiki.objects.DBStringListProperty" class and it stores the data for XWiki objects of type "Database List". The data for list items is stored in the `xwikilistitems` table. Also, the "com.xpn.xwiki.objects.DBStringListProperty" class is a sub-class of the "com.xpn.xwiki.objects.BaseProperty" class.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseProperty" table="xwikiproperties">
 <composite-id unsaved-value="undefined">
  <key-property name="id" column="XWP_ID" type="long" />
  <key-property name="name" type="string">
   <column name="XWP_NAME" index="PROP_NAME" />
  </key-property>
 </composite-id>
 <property name="classType" type="string" column="XWP_CLASSTYPE" length="255" />
 <joined-subclass name="com.xpn.xwiki.objects.DBStringListProperty" table="xwikilists">
  <key>
   <column name="XWL_ID" />
   <column name="XWL_NAME" index="XWLIST_NAME" />
  </key>
  <list name="list" table="xwikilistitems" lazy="false">
   <key>
    <column name="XWL_ID" />
    <column name="XWL_NAME" index="XWLI_NAME" />
   </key>
   <index column="XWL_NUMBER" />
   <element type="string">
    <column name="XWL_VALUE" index="XWLI_VALUE" />
   </element>
  </list>
 </joined-subclass>
</class>
```

The columns of the `xwikilists` table are:

Column	Data type	Default value	not-null	Index
XWL_ID (primary key)	bigint(20)	0	true	-
XWL_NAME (primary key)	varchar(255)	-	true	XWLIST_NAME

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikilongs" Table
  - The "xwikilistitems" Table
  - The "xwikilargestrings" Table
  - The "xwikiintegers" Table
  - The "xwikifloats" Table
  - The "xwikidoubles" Table
  - The "xwikidates"Table

The "xwikilock" Table

The `xwikilock` table corresponds to the "com.xpn.xwiki.doc.XWikiLock" class and it stores information about locks on document edition.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.doc.XWikiLock" table="xwikilock">
 <id name="docId" type="long" unsaved-value="undefined">
  <column name="XWL_DOC_ID" not-null="true" />
  <generator class="assigned" />
 </id>
 <property name="userName" type="string" column="XWL_AUTHOR" length="255" />
 <property name="date" type="timestamp" column="XWL_DATE" not-null="true" />
</class>
```

The columns of the `xwikilock` table are:

Column	Data type	"XWikiLock" class property	Default value	not-null	Index
XWL_DOC_ID (primary key)	bigint(20)	-	0	true	-
XWL_AUTHOR	varchar(255)	userName	null	false	-
XWL_DATE	datetime	date	-	true	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - XWiki Database Schema
  - The "xwikidbversion" Table
  - Programming Overview

The "xwikilongs" Table

The `xwikilongs` table stores data from XWiki object properties of type "Long" and it corresponds to the "com.xpn.xwiki.objects.LongProperty" class which is a sub-class of the "com.xpn.xwiki.objects.BaseProperty" class.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseProperty" table="xwikiproperties">
 <composite-id unsaved-value="undefined">
  <key-property name="id" column="XWP_ID" type="long" />
  <key-property name="name" type="string">
   <column name="XWP_NAME" index="PROP_NAME" />
  </key-property>
 </composite-id>
 <property name="classType" type="string" column="XWP_CLASSTYPE" length="255" />
 <joined-subclass name="com.xpn.xwiki.objects.LongProperty" table="xwikilongs">
  <key>
   <column name="XWL_ID" />
   <column name="XWL_NAME" index="XWLONG_NAME" />
  </key>
  <property name="value" type="long" column="XWL_VALUE" index="XWLONG_VALUE" />
 </joined-subclass>
</class>
```

The columns of the `xwikilongs` table are:

Column	Data type	"LongProperty" class property	Default value	not-null	Index
XWL_ID (primary key)	bigint(20)	-	0	true	-
XWL_NAME (primary key)	varchar(255)	-	-	true	XWLONG_NAME
XWL_VALUE	bigint(20)	value	null	false	XWLONG_VALUE

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikilists" Table
  - The "xwikilistitems" Table
  - The "xwikilargestrings" Table
  - The "xwikiintegers" Table
  - The "xwikifloats" Table
  - The "xwikidoubles" Table
  - The "xwikidates"Table

The "xwikiobjects" Table

The `xwikiobjects` table corresponds to the "com.xpn.xwiki.objects.BaseObject" class and it stores the list of XWiki Objects attached to wiki pages. The actual data from the object properties is stored in the dedicated table, depending on the data type.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseObject" table="xwikiobjects">
 <id name="id" type="long" unsaved-value="undefined">
  <column name="XWO_ID" not-null="true" />
  <generator class="assigned" />
 </id>
 <property name="number" type="integer" column="XWO_NUMBER" index="OBJ_NUMBER" not-null="false" />
 <property name="name" type="string" column="XWO_NAME" index="OBJ_NAME" not-null="true" length="255" />
 <property name="className" type="string" column="XWO_CLASSNAME" index="OBJ_CLASSNAME" not-null="true"
length="255" />
 <property name="guid" type="string" column="XWO_GUID" not-null="false" length="255" />
</class>
```

The columns of the `xwikiobjects` table are:

Column	Data type	"BaseObject" class property	Default value	not-null	Index
XWO_ID (primary key)	bigint(20)	-	0	true	-
XWO_NUMBER	int(11)	number	0	false	OBJ_NUMBER
XWO_NAME	varchar(255)	name	-	true	OBJ_NAME
XWO_CLASSNAME	varchar(255)	className	-	true	OBJ_CLASSNAME
XWO_GUID	varchar(255)	guid	-	false	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikiproperties" Table
  - The "xwikipreferences" Table
  - The "xwikilongs" Table
  - The "xwikilists" Table
  - The "xwikilistitems" Table
  - The "xwikilargestrings" Table
  - The "xwikiintegers" Table
  - The "xwikifloats" Table
  - The "xwikidoubles" Table
  - The "xwikidates"Table
  - The "xwikicomments" Table

The "xwikipreferences" Table

The `xwikipreferences` table stores data from "XWiki.XWikiPreferences" objects.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<!-- XWikiPreferences custom mapping -->
<class entity-name="XWiki.XWikiPreferences" table="xwikipreferences">
 <id name="id" type="long" unsaved-value="undefined">
  <column name="XWP_ID" not-null="true" />
  <generator class="assigned" />
 </id>
 <property name="language" type="string" column="XWP_LANGUAGE" length="255" />
 <property name="default_language" type="string" column="XWP_DEFAULT_LANGUAGE" length="255" />
 <property name="multilingual" type="integer" column="XWP_MULTI_LINGUAL" />
 <property name="authenticate_edit" type="integer" column="XWP_AUTHENTICATE_EDIT" />
 <property name="authenticate_view" type="integer" column="XWP_AUTHENTICATE_VIEW" />
 <property name="auth_active_check" type="integer" column="XWP_AUTH_ACTIVE_CHECK" />
 <property name="backlinks" type="integer" column="XWP_BACKLINKS" />
 <property name="skin" type="string" column="XWP_SKIN" length="255" />
 <property name="stylesheet" type="string" column="XWP_STYLESHEET" length="255" />
 <property name="stylesheets" type="string" column="XWP_STYLESHEETS" length="255" />
 <property name="editor" type="string" column="XWP_EDITOR" length="255" />
 <property name="webcopyright" type="string" column="XWP_WEBCOPYRIGHT" length="255" />
 <property name="title" type="string" column="XWP_TITLE" length="255" />
 <property name="version" type="string" column="XWP_VERSION" length="255" />
 <property name="meta" type="text" column="XWP_META" length="30000" />
 <property name="use_email_verification" type="integer" column="XWP_USE_EMAIL_VERIFICATION" />
 <property name="smtp_server" type="string" column="XWP_SMTP_SERVER" length="255" />
 <property name="admin_email" type="string" column="XWP_ADMIN_EMAIL" length="255" />
 <property name="validation_email_content" type="text" column="XWP_VALIDATION_EMAIL_CONTENT"
length="30000" />
 <property name="confirmation_email_content" type="text" column="XWP_CONFIRMATION_EMAIL_CONTENT"
length="30000" />
 <property name="invitation_email_content" type="text" column="XWP_INVITATION_EMAIL_CONTENT"
length="30000" />
 <property name="leftPanels" type="string" column="XWP_LEFT_PANELS" length="2000" />
 <property name="rightPanels" type="string" column="XWP_RIGHT_PANELS" length="2000" />
 <property name="showLeftPanels" type="integer" column="XWP_SHOW_LEFT_PANELS" />
 <property name="showRightPanels" type="integer" column="XWP_SHOW_RIGHT_PANELS" />
 <property name="languages" type="string" column="XWP_LANGUAGES" length="255" />
 <property name="registration_anonymous" type="string" column="XWP_REGISTRATION_ANONYMOUS" length="255"
/>
 <property name="registration_registered" type="string" column="XWP_REGISTRATION_REGISTERED" length="255" />
 <property name="edit_anonymous" type="string" column="XWP_EDIT_ANONYMOUS" length="255" />
 <property name="edit_registered" type="string" column="XWP_EDIT_REGISTERED" length="255" />
 <property name="comment_anonymous" type="string" column="XWP_COMMENT_ANONYMOUS" length="255" />
 <property name="comment_registered" type="string" column="XWP_COMMENT_REGISTERED" length="255" />
 <property name="documentBundles" type="string" column="XWP_DOCUMENT_BUNDLES" length="2000" />
</class>
```

The columns of the `xwikipreferences` table are:

Column	Data type	"XWiki.XWiki Preferences" class property	Default value	not-null	Index
XWP_ID (primary key)	bigint(20)	-	0	true	-
XWP_LANGUAGE	varchar(255)	language	null	false	-
XWP_DEFAULT _LANGUAGE	varchar(255)	default_language	null	false	-
XWP_MULTI _LINGUAL	int(11)	multilingual	0	-	-
XWP	int(11)	authenticate_edit	0	-	-

_AUTHENTICATE_EDIT					
XWP_AUTHENTICATE_VIEW	int(11)	authenticate_view	0	-	-
XWP_AUTH_ACTIVE_CHECK	int(11)	auth_active_check	0	-	-
XWP_BACKLINKS	int(11)	backlinks	0	-	-
XWP_SKIN	varchar(255)	skin	null	false	-
XWP_STYLESHEET	varchar(255)	stylesheet	null	false	-
XWP_STYLESHEETS	varchar(255)	stylesheets	null	false	-
XWP_EDITOR	varchar(255)	editor	null	false	-
XWP_WEBCOPYRIGHT	varchar(255)	webcopyright	null	false	-
XWP_TITLE	varchar(255)	title	null	false	-
XWP_VERSION	varchar(255)	version	null	false	-
XWP_META	longtext	meta	null	false	-
XWP_USE_EMAIL_VERIFICATION	int(11)	use_email_verification	0	-	-
XWP_SMTP_SERVER	varchar(255)	smtp_server	null	false	-
XWP_ADMIN_EMAIL	varchar(255)	admin_email	null	false	-
XWP_VALIDATION_EMAIL_CONTENT	longtext	validation_email_content	null	false	-
XWP_CONFIRMATION_EMAIL_CONTENT	longtext	confirmation_email_content	null	false	-
XWP_INVITATION_EMAIL_CONTENT	longtext	invitation_email_content	null	false	-
XWP_LEFT_PANELS	varchar(2000)	leftPanels	null	false	-
XWP_RIGHT_PANELS	varchar(2000)	rightPanels	null	false	-
XWP_SHOW_LEFT_PANELS	int(11)	showLeftPanels	0	-	-
XWP_SHOW_RIGHT_PANELS	int(11)	showRightPanels	0	-	-
XWP_LANGUAGES	varchar(255)	languages	null	false	-
XWP_REGISTRATION_ANONYMOUS	varchar(255)	registration_anonymous	null	false	-
XWP_REGISTRATION_REGISTERED	varchar(255)	registration_registered	null	false	-
XWP_EDIT_ANONYMOUS	varchar(255)	edit_anonymous	null	false	-
XWP_EDIT_REGISTERED	varchar(255)	edit_registered	null	false	-
XWP_COMMENT_ANONYMOUS	varchar(255)	comment_anonymous	null	false	-
XWP_COMMENT_REGISTERED	varchar(255)	comment_registered	null	false	-
XWP_DOCUMENT_BUNDLES	varchar(2000)	documentBundles	null	false	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikicomments" Table

The "xwikiproperties" Table

The `xwikiproperties` table corresponds to the "com.xpn.xwiki.objects.BaseProperty" class and it stores the list of properties of XWiki objects attached to wiki pages. The values of the properties are stored in the corresponding tables, depending on the data type:

Type	Java Class	Table
Integer	com.xpn.xwiki.objects.IntegerProperty	xwikiintegers
Long	com.xpn.xwiki.objects.LongProperty	xwikilongs
Float	com.xpn.xwiki.objects.FloatProperty	xwikifloats
Double	com.xpn.xwiki.objects.DoubleProperty	xwikidoubles
String	com.xpn.xwiki.objects.StringProperty	xwikistrings
Date	com.xpn.xwiki.objects.DateProperty	xwikidates
Text Area	com.xpn.xwiki.objects.LargeStringProperty	xwikilargestrings
Static List	com.xpn.xwiki.objects.StringListProperty	xwikilargestrings
Database List	com.xpn.xwiki.objects.DBStringListProperty	xwikilists

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseProperty" table="xwikiproperties">
 <composite-id unsaved-value="undefined">
  <key-property name="id" column="XWP_ID" type="long" />
  <key-property name="name" type="string">
   <column name="XWP_NAME" index="PROP_NAME" />
  </key-property>
 </composite-id>
 <property name="classType" type="string" column="XWP_CLASSTYPE" length="255" />
</class>
```

The columns of the `xwikiproperties` table are:

Column	Data type	"BaseProperty" class property	Default value	not-null	Index
XWP_ID (primary key)	bigint(20)	id	0	true	-
XWP_NAME (primary key)	varchar(255)	name	-	true	PROP_NAME
XWP_CLASSTYPE	varchar(255)	classType	-	-	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistrings" Table
  - The "xwikipreferences" Table
  - The "xwikiobjects" Table
  - The "xwikilongs" Table
  - The "xwikilists" Table
  - The "xwikilistitems" Table
  - The "xwikilargestrings" Table
  - The "xwikiintegers" Table
  - The "xwikifloats" Table
  - The "xwikidoubles" Table
  - The "xwikidates"Table
  - The "xwikicomments" Table

The "xwikircs" Table

The `xwikircs` table corresponds to the "com.xpn.xwiki.doc.rcs.XWikiRCSNodeInfo" and "com.xpn.xwiki.doc.rcs.XWikiRCSNodeContent" classes and it stores information about XWiki documents versions.

To better explain the reason why we need 2 classes, we will take the "History" tab as example. Normally, when displaying the list of available versions, you see information like the version number, the last editor, the date of the last modification and a summary explaining what changed since the last version. In this case, XWiki only needs to load the "XWikiRCSNodeInfo" class. On the other hand, when comparing 2 versions, a diff is displayed, which requires the loading of the second class, "XWikiRCSNodeContent".

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.doc.rcs.XWikiRCSNodeInfo" table="xwikircs">
 <composite-id name="id" class="com.xpn.xwiki.doc.rcs.XWikiRCSNodeId">
  <key-property name="docId" column="XWR_DOCID" type="long" />
  <key-property name="version1" column="XWR_VERSION1" type="integer" />
  <key-property name="version2" column="XWR_VERSION2" type="integer" />
 </composite-id>
 <property name="date" type="timestamp" column="XWR_DATE" index="REV_DATE" not-null="true" />
 <!-- Note: Length must be the same one as XWD_COMMENT since it's the same content being put in history -->
 <property name="comment" type="string" column="XWR_COMMENT" length="1023" not-null="true" />
 <!-- Note: Length must be the same one as XWD_COMMENT since it's the same content being put in history -->
 <property name="author" type="string" column="XWR_AUTHOR" length="255" index="REV_AUTHOR" not-null="true"
/>
 <property name="diff" type="boolean" column="XWR_ISDIFF" index="REV_ISDIFF" not-null="false" update="false"
insert="false" />
</class>
<class name="com.xpn.xwiki.doc.rcs.XWikiRCSNodeContent" table="xwikircs">
 <composite-id name="id" class="com.xpn.xwiki.doc.rcs.XWikiRCSNodeId">
  <key-property name="docId" column="XWR_DOCID" type="long" />
  <key-property name="version1" column="XWR_VERSION1" type="integer" />
  <key-property name="version2" column="XWR_VERSION2" type="integer" />
 </composite-id>
 <component name="patch">
  <property name="diff" type="boolean" column="XWR_ISDIFF" not-null="false" />
  <property name="content" type="materialized_clob" column="XWR_PATCH" not-null="false" length="1000000000" />
 </component>
</class>
```

The columns of the `xwikircs` table are:

Column	Data type	"XWikiRCSNode Info" class property	Default value	not-null	Index
XWR_DOCID (primary key)	bigint(20)	docId	0	true	-
XWR_VERSION1 (primary key)	int(11)	version1	0	true	-
XWR_VERSION2 (primary key)	int(11)	version2	0	true	-
XWR_DATE	datetime	date	0000-00-00 00:00:00	true	REV_DATE
XWR_COMMENT	varchar(1023)	comment	-	true	-
XWR_AUTHOR	varchar(255)	author	-	true	REV_AUTHOR
XWR_ISDIFF	bit(1)	diff	-	false	REV_ISDIFF
Column	**Data type**	**"XWikiRCSNode Content" class property**	**Default value**	**not-null**	**Index**
XWR_DOCID (primary key)	bigint(20)	docId	0	true	-
XWR_VERSION1 (primary key)	int(11)	version1	0	true	-
XWR_VERSION2 (primary key)	int(11)	version2	0	true	-

XWR_ISDIFF	bit(1)	diff	-	false	REV_ISDIFF
XWR_PATCH	longtext	content	-	false	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **User Guide**
  - XWiki Watchlist Application
  - Version Control
  - Page History
  - Common Edit Actions
- **Programming Guide**
  - The "xwikiattrecyclebin" Table
  - The "xwikiattachment_content" Table
  - The "xwikiattachment_archive" Table
  - The "xwikiattachment" Table
- **Admin Guide**
  - Import

The "xwikirecyclebin" Table

The `xwikirecyclebin` table corresponds to the "com.xpn.xwiki.doc.XWikiDeletedDocument" class and it stores the list of documents deleted from the wiki.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.doc.XWikiDeletedDocument" table="xwikirecyclebin" mutable="false">
 <id name="id" column="XDD_ID">
  <generator class="native" />
 </id>
 <natural-id mutable="false">
  <property name="fullName" column="XDD_FULLNAME" type="string" length="255" />
  <property name="language" column="XDD_LANGUAGE" type="string" length="5" index="XDD_LANGUAGE" />
  <property name="date" type="timestamp" column="XDD_DATE" index="XDD_DATE" />
 </natural-id>
 <property name="deleter" type="string" column="XDD_DELETER" index="XDD_DELETER" />
 <property name="xml" type="materialized_clob" column="XDD_XML" not-null="true" length="1000000000" />
</class>
```

The columns of the `xwikirecyclebin` table are:

Column	Data type	"XWikiDeleted Document" class property	Default value	not-null	Index
XDD_ID (primary key, auto incremented)	bigint(20)	-	0	true	-
XDD_FULLNAME (unique)	varchar(255)	fullName	-	true	-
XDD_LANGUAGE (unique)	varchar(5)	language	-	true	XDD_LANGUAGE
XDD_DATE (unique)	datetime	date	0000-00-00 00:00:00	true	XDD_DATE
XDD_DELETER	varchar(255)	deleter	null	false	XDD_DELETER
XDD_XML	longtext	xml	-	true	-

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikilinks" Table
  - The "xwikidoc" Table
  - The "xwikiattrecyclebin" Table
  - The "xwikiattachment_content" Table
  - The "xwikiattachment_archive" Table
  - The "xwikiattachment" Table

The "xwikistatsdoc" Table

The `xwikistatsdoc` table corresponds to the "com.xpn.xwiki.stats.impl.DocumentStats" class and it stores information about XWiki pages and spaces view and edit statistics.

In case you haven't enabled the Statistics plugin on your wiki, the `xwikistatsdoc` table will be empty.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.stats.impl.DocumentStats" table="xwikistatsdoc">
 <id name="id" type="long" unsaved-value="undefined">
  <column name="XWS_ID" not-null="true" />
  <generator class="assigned" />
 </id>
 <property name="number" type="integer" column="XWS_NUMBER" not-null="false" />
 <property name="name" type="string" column="XWS_NAME" not-null="true" length="255" index="XWDS_NAME" />
 <!-- Before XWiki 2.2M2 this column was always containing the "internal" string to tell the storage
      implementation Stats objects were special. Starting with XWiki 2.2M2 the check is now on whether the class
      name is empty ("") or null. Thus this column can now contain both "internal" (old stats) and ""
      (new stats) -->
 <property name="className" type="string" column="XWS_CLASSNAME" length="255" />
 <property name="action" type="string" column="XWS_ACTION" not-null="true" length="255" index="XWDS_ACTION"
/>
 <property name="pageViews" type="integer" column="XWS_PAGE_VIEWS" index="XWDS_PAGE_VIEWS" not-
null="false" />
 <property name="uniqueVisitors" type="integer" column="XWS_UNIQUE_VISITORS"
index="XWDS_UNIQUE_VISITORS" not-null="false" />
 <property name="period" type="integer" column="XWS_PERIOD" index="XWDS_PERIOD" not-null="false" />
 <property name="visits" type="integer" column="XWS_VISITS" index="XWDS_VISITS" not-null="false" />
</class>
```

The columns of the `xwikistatsdoc` table are:

Column	Data type	"DocumentStats" class property	Default value	not-null	Index
XWS_ID (primary key)	bigint(20)	-	0	true	-
XWS_NUMBER	int(11)	number	null	false	-
XWS_NAME	varchar(255)	name	-	true	XWDS_NAME
XWS_CLASSNAME	varchar(255)	className	-	-	-
XWS_ACTION	varchar(255)	action	-	true	XWDS_ACTION
XWS_PAGE_VIEWS	int(11)	pageViews	null	false	XWDS_PAGE_VIEWS
XWS_UNIQUE_VISITORS	int(11)	uniqueVisitors	null	false	XWDS_UNIQUE_VISITORS
XWS_PERIOD	int(11)	period	null	false	XWDS_PERIOD
XWS_VISITS	int(11)	visits	null	false	XWDS_VISITS

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistatsvisit" Table
  - The "xwikistatsreferer" Table
- **Admin Guide**
  - XWiki Statistics Application
  - Activate and Configure Google Analytics

The "xwikistatsreferer" Table

The `xwikistatsreferer` table corresponds to the "com.xpn.xwiki.stats.impl.RefererStats" class and it stores incoming external links on the wiki.

In case you haven't enabled the Statistics plugin on your wiki, the `xwikistatsreferer` table will be empty.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.stats.impl.RefererStats" table="xwikistatsreferer">
 <id name="id" type="long" unsaved-value="undefined">
  <column name="XWR_ID" not-null="true" />
  <generator class="assigned" />
 </id>
 <property name="number" type="integer" column="XWR_NUMBER" not-null="false" />
 <property name="name" type="string" column="XWR_NAME" not-null="true" length="255" index="XWRS_NAME" />
 <!-- Before XWiki 2.2M2 this column was always containing the "internal" string to tell the storage
      implementation Stats objects were special. Starting with XWiki 2.2M2 the check is now on whether the class
      name is empty ("") or null. Thus this column can now contain both "internal" (old stats) and ""
     (new stats) -->
 <property name="className" type="string" column="XWR_CLASSNAME" length="255" />
 <property name="referer" type="string" column="XWR_REFERER" not-null="true" length="8192" />
 <property name="pageViews" type="integer" column="XWR_PAGE_VIEWS" index="XWRS_PAGE_VIEWS" not-
null="false" />
 <property name="period" type="integer" column="XWR_PERIOD" index="XWRS_PERIOD" not-null="false" />
</class>
```

The columns of the `xwikistatsreferer` table are:

Column	Data type	"RefererStats" class property	Default value	not-null	Index
XWR_ID (primary key)	bigint(20)	-	0	true	-
XWR_NUMBER	int(11)	number	null	false	-
XWR_NAME	varchar(255)	name	-	true	XWRS_NAME
XWR_CLASSNAME	varchar(255)	className	-	-	-
XWR_REFERER	varchar(8192)	referer	-	true	-
XWR_PAGE_VIEWS	int(11)	pageViews	-	false	XWRS_PAGE_VIEWS
XWR_PERIOD	int(11)	period	-	false	XWRS_PERIOD

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistatsvisit" Table
  - The "xwikistatsdoc" Table
- **Admin Guide**
  - XWiki Statistics Application
  - Activate and Configure Google Analytics

The "xwikistatsvisit" Table

The `xwikistatsvisit` table corresponds to the "com.xpn.xwiki.stats.impl.VisitStats" class and it stores data about the users sessions.

In case you haven't enabled the Statistics plugin on your wiki, the `xwikistatsvisit` table will be empty.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.stats.impl.VisitStats" table="xwikistatsvisit">
 <id name="id" type="long" unsaved-value="undefined">
  <column name="XWV_ID" not-null="true" />
  <generator class="assigned" />
 </id>
 <property name="number" type="integer" column="XWV_NUMBER" not-null="false" />
 <property name="name" type="string" column="XWV_NAME" not-null="true" length="255" index="XWVS_NAME" />
<!-- Before XWiki 2.2M2 this column was always containing the "internal" string to tell the storage
      implementation Stats objects were special. Starting with XWiki 2.2M2 the check is now on whether the class
      name is empty ("") or null. Thus this column can now contain both "internal" (old stats) and ""
      (new stats) -->
 <property name="className" type="string" column="XWV_CLASSNAME" length="255" />
 <property name="IP" type="string" column="XWV_IP" not-null="true" length="255" index="XWVS_IP" />
 <property name="userAgent" type="string" column="XWV_USER_AGENT" not-null="true" length="8192" />
 <property name="cookie" type="string" column="XWV_COOKIE" not-null="true" length="8192" />
 <property name="uniqueID" type="string" column="XWV_UNIQUE_ID" index="XWVS_UNIQUE_ID" not-null="true"
length="255" />
 <property name="pageViews" type="integer" column="XWV_PAGE_VIEWS" index="XWVS_PAGE_VIEWS" not-
null="false" />
 <property name="pageSaves" type="integer" column="XWV_PAGE_SAVES" index="XWVS_PAGE_SAVES" not-
null="false" />
 <property name="downloads" type="integer" column="XWV_DOWNLOADS" index="XWVS_DOWNLOADS" not-
null="false" />
 <property name="startDate" type="timestamp" column="XWV_START_DATE" index="XWVS_START_DATE" not-
null="false" />
 <property name="endDate" type="timestamp" column="XWV_END_DATE" index="XWVS_END_DATE" not-null="false"
/>
</class>
```

The columns of the `xwikistatsvisit` table are:

Column	Data type	"VisitStats" class property	Default value	not-null	Index
XWV_ID (primary key)	bigint(20)	-	0	null	-
XWV_NUMBER	int(11)	number	null	false	-
XWV_NAME	varchar(255)	name	-	true	XWVS_NAME
XWV_CLASSNAME	varchar(255)	className	-	-	-
XWV_IP	varchar(255)	IP	-	true	XWVS_IP
XWV_USER_AGENT	varchar(8192)	userAgent	-	true	-
XWV_COOKIE	varchar(8192)	cookie	-	true	-
XWV_UNIQUE_ID	varchar(255)	uniqueID	-	true	XWVS_UNIQUE_ID
XWV_PAGE_VIEWS	int(11)	pageViews	null	false	XWVS_PAGE_VIEWS
XWV_PAGE_SAVE	int(11)	pageSaves	null	false	XWVS_PAGE_SAVES
XWV_DOWNLOADS	int(11)	downloads	null	false	XWVS_DOWNLOADS
XWV_START_DATE	datetime	startDate	null	false	XWVS_START_DATE
XWV_END_DATE	datetime	endDate	null	false	XWVS_END_DATE

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikistatsreferer" Table
  - The "xwikistatsdoc" Table
- **Admin Guide**
  - XWiki Statistics Application
  - Activate and Configure Google Analytics

The "xwikistrings" Table

The `xwikistrings` table stores data from XWiki object properties of type "String" (short text) and it corresponds to the "com.xpn.xwiki.objects.StringProperty" class which is a sub-class of the "com.xpn.xwiki.objects.BaseProperty" class.

The mapping information is available in the `xwiki.hbm.xml` file:

```
<class name="com.xpn.xwiki.objects.BaseProperty" table="xwikiproperties">
 <composite-id unsaved-value="undefined">
  <key-property name="id" column="XWP_ID" type="long" />
  <key-property name="name" type="string">
   <column name="XWP_NAME" index="PROP_NAME" />
  </key-property>
 </composite-id>
 <property name="classType" type="string" column="XWP_CLASSTYPE" length="255" />
 <joined-subclass name="com.xpn.xwiki.objects.StringProperty" table="xwikistrings">
  <key>
   <column name="XWS_ID" />
   <column name="XWS_NAME" index="XWSTR_NAME" />
  </key>
  <property name="value" type="string" column="XWS_VALUE" length="255" index="XWSTR_VALUE" />
 </joined-subclass>
</class>
```

The columns of the `xwikistrings` table are:

Column	Data type	"StringProperty" class property	Default value	not-null	Index
XWS_ID (primary key)	bigint(20)	-	0	true	-
XWS_NAME (primary key)	varchar(255)	-	-	true	XWSTR_NAME
XWS_VALUE	varchar(255)	value	-	-	XWSTR_VALUE

To know more about the XWiki database schema, go to the dedicated documentation page.

Related Pages

- **Programming Guide**
  - The "xwikiproperties" Table
  - The "xwikiobjects" Table
  - The "xwikilongs" Table
  - The "xwikilists" Table
  - The "xwikilistitems" Table
  - The "xwikilargestrings" Table
  - The "xwikiintegers" Table
  - The "xwikifloats" Table
  - The "xwikidoubles" Table
  - The "xwikidates"Table

XWiki Query Guide

All data making up the XWiki application can be queried using Query languages. The supported Query languages are:

- XWiki Query Language (XWQL)
- Hibernate Query Language (HQL)

At first, XWiki was only supporting HQL but that led to complex queries in some instances and thus we have introduced XWQL. This is a JPQL superset which makes it much easier to write queries for the XWiki model, as you can see in the examples below.

XWiki-Specific Extensions in XWQL over JPQL

- Short form queries:
  - `where <expr>` means `select doc.fullName where <expr>`
  - `from <fromlist> [where <expr>]` means `select doc.fullName from Document as doc, <fromlist> [where <expr>]`
- Special syntax for XWiki objects in `from` and `where` clauses:
  - `from doc.object(Class) as obj`
  - `where doc.object(Class).prop = 'something'`

Query Language Examples

All queries starting with `select` require programming rights to execute.

Query Description	XWQL	HQL
Query listing all documents	<empty query>	<empty query>
Query listing all documents created after a given date	`where doc.creationDate > '2008-01-01'`	`where doc.creationDate > '2008-01-01'`
Query listing all documents last updated by a given user	`where doc.author = 'XWiki.LudovicDubost'`	`where doc.author = 'XWiki.LudovicDubost'`
Query listing all documents containing XWiki Objects (XObject) of a given XWiki Class (XClass)	`from doc.object(XWiki.XWikiUsers) as user`	`, BaseObject as obj where doc.fullName = obj.name and obj.className = 'XWiki.XWikiUsers'`
Query on XObjects and filtering on XObject property value	`where doc.author = 'XWiki.LudovicDubost' and doc.object(XWiki.XWikiUsers) .email like '%xwiki.com'`	`, BaseObject as obj, StringProperty as prop where doc.fullName = obj.name and obj.className = 'XWiki.XWikiUsers' and obj.id=prop.id.id and prop.id.name='email' and prop.value like '%xwiki.com' and doc.author ='XWiki.LudovicDubost'`
Query on 2 XObjects	`where doc.object(XWiki.XWikiUsers) .email like '%xwiki.com' and doc.object(XWiki.ArticleClass) .content like '%ludovic%'`	`, BaseObject as obj, StringProperty as prop, BaseObject as obj2, LargeStringProperty as contentprop where doc.fullName = obj.name and obj.className = 'XWiki.XWikiUsers' and obj.id=prop.id.id and prop.id.name='email' and prop.value like '%xwiki.com'`

		`and doc.fullName=obj2.name and obj2.className='XWiki.ArticleClass' and obj2.id=contentprop.id.id and contentprop.id.name='content' and contentprop.value like '%ludovic%'`
Search blogs per category	`where doc.fullName <> 'XWiki.ArticleClassTemplate' and :category member of doc.object(XWiki.ArticleClass) .category`	`, BaseObject as obj, DBStringListProperty as prop join prop.list list where obj.name=doc.fullName and obj.className='XWiki.ArticleClass' and obj.name<>'XWiki.ArticleClassTemplate' and obj.id=prop.id.id and prop.id.name='category' and list='${category}' order by doc.creationDate desc`
List all tags	`select distinct obj.tags from Document doc, doc.object(XWiki.TagClass) as obj`	`select distinct tag from BaseObject as obj, DBStringListProperty as prop join prop.list as tag where obj.className='XWiki.TagClass' and obj.id=prop.id.id and prop.id.name='tags'`

How to Perform Queries

Using the Query Manager

The principle is to get a reference to the **Query Manager**, then call a method on it to create a query for the given query language. It is possible to set some behaviors on the Query object like:

- limiting the number of the returned result
- binding variables
- setting the offset
- setting the wiki on which to execute the query

From Velocity

```
// For XWQL
$xwiki.queryManager.xwql(xwqlstatement).execute()
// For HQL
$xwiki.queryManager.createQuery(hqlstatement, "hql").execute()
```

From Other Scripting Languages

For XWQL (example with Groovy):

```
services.get("query").xwql(xwqlstatement).execute()
```

For HQL:

```
services.get("query").hql(hqlstatement).execute()
```

From Java Components

Get a `QueryManager` injected. For example:

```
/**
 * Secure query manager that performs checks on rights depending on the query being executed.
 */
@Inject
@Named("secure")
private QueryManager queryManager;
```

Obtain and execute a query:

```
this.queryManager.createQuery(statement, Query.XWQL).execute();
```

Non-exhaustive List of Queryable Object Fields

XWikiDocument

- **XWikiDocument.fullName** : full name, including space and page name.
  - Example value: `Main.WebHome`
- **XWikiDocument.author** : last editor.
  - Example value: `XWiki.Admin`
- **XWikiDocument.creator** : first editor.
  - Example value: `XWiki.Admin`

BaseObject

- **BaseObject.id** : arbitrary unique id of the object.
  - Example value: `123456789`
- **BaseObject.className** : class.
  - Example value: `XWiki.XWikiUsers`

*Property (StringProperty, IntegerProperty, etc)

- **Property.id.id** : unique id of the object the property belongs to.
  - Example value: `123456789`
- **Property.name** : name of the property.
  - Example value: `first_name`
- **Property.value** : value.
  - Example value: `John`

Related Pages

- **Programming Guide**
  - XWiki Scripting
  - XWiki Database Schema
  - XWiki API Reference
  - Solr Schema and API
  - Programming Overview
  - Livetable Macro
  - Documents Macro
- **Admin Guide**
  - Customize the Search Feature

Solr Schema and API

Solr is the XWiki Enterprise default search engine and it is based on Apache Solr. Solr supports full-text search, hit highlighting, faceted search, dynamic clustering, database integration, rich document handling, and geospatial search. The plugin architecture allows the setup of different type of analyzers using XML configuration files.

Solr Schema

According to the XWiki Data Model, the supported entities are:

- wiki
- space
- page
- class
- object
- object property
- attachment

Entity types have field they share, as well as specific fields.

Fields Shared by All Indexed Entities

The `wiki`, `space` and `name` information is shared because each indexed entity is either a document or held by a document. According to the Solr schema, these fields are:

Name	Description
id	A keyword field that holds a unique string which identifies a document across the index. This field is used for finding old versions of a document to be indexed.
type	The type of entity to be indexed: `DOCUMENT`, `ATTACHMENT`, `OBJECT`, `OBJECT_PROPERTY`.
wiki	A keyword field holding the name of the virtual wiki a document belongs to.
space	The name of the space the document belongs to. This field is analyzed and used for free text search.
space_exact	The unanalyzed and not stored version of the document's space.
name	The name of the document. This field is analyzed and mostly used for free text search.
name_exact	The unanalyzed and not stored version of the document's name.
locale	The real/calculated locale of the document i.e. the default locale in default document entry case.
locales	The list of Locales covered by the entity. The list is dynamically determined from the list of enabled Locales and the various Locales of the associated wiki document.
language	The language of the document.
hidden	A document hidden flag. Only documents can be made hidden explicitly because attachments, objects and object properties are automatically hidden if the document that holds them is hidden.

Document Static Fields

Name	Description
fullname	The document full name: `SpaceName.PageName`.
title	A multilingual and virtual field representing the document title.
title_	The localized title which is indexed based on the document locale. E.g. `title_en`.
title_sort	The dedicated field for sort which is necessary because analyzed fields cannot be used for sorting.
doccontent_	The rendered document content i.e. with no transformations executed (e.g. `doccontent_pt_BR`). This allows to use a different boost value for document content than for the object (`objcontent`) and the attachment content (`attcontent`).
doccontentraw_	
version	The document version which needs to be indexed in order to be able to detect whether the index is synched with the database.
comment_	The version summary. E.g. `comment_en`.
doclocale	The technical locale of the document.
author	The last author. This field is used for faceting.
author_display	The last analyzed author. This field is used for free text search.
author_display_sort	The dedicated field used for sorting.
creator	The document creator, stored verbatim (unanalyzed) for faceting.
creator_display	The analyzed document creator, used for free text search.
date	
creationdate	

In order to be able to mix the single entity approach with the multiple entities one and to avoid joins, we have to duplicate information. This means that for each entity we have to index and duplicate information about other related entities.

Object Data

Name	Description
class	The type of objects stored by the document. E.g. `XWiki.TagClass`. You can also use `object` in search queries as an alias to `class`.
objcontent_	A collection of values from all the properties of all the objects found on the indexed document.
property.spaceName.className.propertyName_sort*	A dedicated field for sorting on property values, which is needed because Sol doesn't support sorting on multivalued fields. E.g. `property.Blog.BlogPostClass.publishDate_sortDate`.
object.spaceName.className_	A dynamic multivalued field indexing the entire content of the objects of the specified type. All values are indexed as localized text, using the document locale. E.g. `object.Blog.BlogPostClass_fr`.
property.spaceName.className.propertyName_	A dynamic multivalued field indexing the value of the specified property. For static lists, both the raw value which is saved in the database and the display value are indexed. Property values are indexed based on their type. E.g.: • `property.Blog.BlogPostClass.published_boolean` • `property.Blog.BlogPostClass.publishDate_date` • `property.Blog.BlogPostClass.category_string` • `property.Blog.BlogPostClass.summary_en`

Attachment Data

Name	Description
filename	The names of the files attached to the document. E.g. `['file.pdf', 'image.jpg']`.
mimetype	A list of attachment media types. E.g. `['text/plain', 'image/png']`.
attauthor	The absolute references of the users that uploaded the last version of each of the document attachments. This field

	is used for faceting. E.g. `['xwiki:XWiki.Admin', 'projects:XWiki.JaneDoe']`.
attauthor_display	The real user names of the users that uploaded the last version of each of the document attachments. This field is used for free text search. E.g. `['Admin', 'Jane Doe']`.
attdate	The dates when the last version of each attachment have been uploaded.
attcontent_	The content of each attachment indexed based on the document locale. E.g. `attcontent_en : ['content of first attachment', 'content of second attachment']`.
attsize	The size of each attachment in bytes.

Attachment Static Fields

Name	Description
filename	The attachment file name. E.g. `['file.pdf']`.
filename_sort	The attachment file name used for sorting.
mimetype	The attachment media type.
attversion	The attachment revision which is used in order to tell whether the Solr index is synched with the database.
attauthor	The absolute reference of the user that uploaded the last version of the attachment. This field is used for faceting. E.g. `['xwiki:XWiki.Admin', 'projects:XWiki.JaneDoe']`.
attauthor_display	The real user name of the user that uploaded the last version of the attachment. This field is used for free text search. E.g. `['Admin', 'Jane Doe']`.
attdate	The date when the last version of the attachment has been uploaded.
attdate_sort	A dedicated field for sorting which is needed because `attadate` is multivalued whereas Solr doesn't support multivalued fields.
attsize	The size of the attachment in bytes.
attsize_sort	A dedicated field for sorting because `attsize` is multivalued.

Object and ObjectProperty Static Fields

Name	Description
class	The object type.
number	The object number which identifies an object when there are multiple objects of the same type on a document.
objcontent_	A collection of the values from all the properties of the indexed object. The used format is `propertyName : propertyValue". This field is analyzed based on the document locale. E.g.` objcontent_ro`.`
property.propertyName_	A dynamic multivalued field indexing the value of the specified property. For static lists, both the raw value which is saved in the database and the display value are indexed. Property values are indexed based on their type. E.g.: • `property.published_boolean` • `property.publishDate_date` • `property.category_string` • `property.summary_en`

Solr Search Query API

The Solr search query API is exposed using the Query Module API.

Common Query Parameters

q

The **q** parameter is the main query for the request. Example: q=foo __INPUT__* bar.

fq

The **fq** parameter stands for Filter Query and it can be used to specify a query for restricting the super set of returned documents without influencing the score.

Queries specified with **fq** are cached independently from the main query. Also, the **fq** parameter can be specified multiple times and documents will only be included in the result if they are in the intersection of the document sets resulting from each **fq**.

Example:
fq=type:ATTACHMENT
fq=wiki:xwiki
fq=space_exact:Main
fq=class:FAQCode.FAQClass

fl

The **fl** parameter is used to specify a set of facet fields to return, limiting the amount of information in the response. The set of fields to be returned can be specified as a space or comma separated list of field names.

The string `score` can also be used to indicate that the score of each document for the particular query should be returned as a field.

The string `*` can be used to indicate all stored fields the document has.

Example:
fl=creator, creationdate, author, date, mimetype, attauthor, attdate, attsize

qf

The **qf** parameter stands for Query Field and it is used to specify in which fields to look as well as their importance. This is actually the query boost.

Example: qf=title^3 property.FAQCode.FAQClass.answer

sort

Sorting can be done:

- by score on any `multiValued="false" indexed="true"` field provided the field is either non-tokenized or it uses an Analyzer that only produces a single term.
- by index id using `sort=_docid_ asc` or `sort=_docid_ desc`.

A sort ordering must include:

- a field name or the pseudo-field `score` followed by
- a white-space escaped as either `+` or `%20` in URL strings followed by
- a sort direction: `asc` or `desc`

Multiple sort orderings can be separated by a comma: `sort=<field name>+<direction>[,<field name>+<direction>]`

Custom Query Parameters

As shown in the solrconfig configuration file, there are 4 available query parameters.

Name	Description	Default Value
xwiki.multilingualFields	Represents the list of multilingual fields that will be expanded in the search query. This way, a user can write a query on the `title` field and all the `title_<language>` variations of the field will be used in the query. The list of languages for which a field is expanded is taken from the `xwiki.supportedLocales` query parameter. If this parameter is not defined, the ROOT locale is used instead.	`title`, `doccontent`, `doccontentraw`, `comment`, `objcontent`, `propertyvalue`, `attcontent`, `property.*`, `object.*`
xwiki.supportedLocales	The list of supported locales used to expand the fields specified by `xwiki.multilingualFields` in search queries.	An empty list which means only the ROOT locale is used for expanding multilingual fields in search queries.

xwiki.typedDynamicFields	The list of typed non-string dynamic fields that will be expanded in the search query. The names of these fields are suffixed with the name of their data type.	`property.*`
xwiki.dynamicFieldTypes	Dynamic field definitions which allow using convention over configuration for fields via the specification of patterns to match field names. The complete list is available in the schema.xml file.	boolean, int, long, float, double, string, date.

Examples

Faceting on Object Properties

To add facets to an "Test.TestClass" object, just use the code below

```
#set ($discard = $query.bindValue('facet.field', ['field2', 'property.Test.TestClass.staticList1_string']))
```

which will be triggered by the query object:Test.TestClass

The _string suffix means the property was indexed verbatim.

Sorting on Object Properties

You can sort the document search results based on a property value using the Query Module API:

```
#set ($discard = $query.bindValue('sort', "property.Test.TestClass.staticList1_sortString asc"))
```

where:

- `Test.TestClass` is the name of the wiki class
- `staticList1` is the name of the "Test.TestClass" property of type Static List
- `sortString` is a suffix representing the dynamic type used for sorting as explained in the above section

Related Pages

- **User Guide**
  - Search Application
- **Programming Guide**
  - XWiki Query Guide
- **Admin Guide**
  - Search Suggest Sources
  - Customize the Search Feature

Groovy Notifications

This tutorial explains how to create a Groovy Event Listener that responds to document create and update events. For this purpose the XWiki Observation module provides APIs which listen to internal XWiki events. The complete list of XWiki Bridge Events is available on GitHub at https://github.com/xwiki/xwiki-platform/blob/master/xwiki-platform-core/xwiki-platform-bridge/src/main/java/org/xwiki/bridge/event/.

A possible use case is the need to create a Groovy Listener that sends an email notification to the farm administrator whenever a page is created or updated in the "Code" space of the main wiki. Also, you might want to keep a log of the new/updated documents in a wiki document, for instance "Admin.Logger". The recommended approach is to create a Scheduler job because this way you can schedule when the event listener sends email notifications. Moreover, there is no risk for the code to be executed after a server restart.

The first step will then be to go to "Scheduler.WebHome" and create a new job, named "Document Updated Event Listener". The Groovy code is:

```
import org.xwiki.observation.*
import org.xwiki.observation.event.*
import org.xwiki.bridge.event.*
import org.xwiki.observation.event.filter.*
import com.xpn.xwiki.web.*
import com.xpn.xwiki.*
class LoggingEventListener implements EventListener
{
  LoggingEventListener()
  {
  }
  String getName()
  {
    // The unique name of this event listener
    return "logging"
  }
  List<Event> getEvents()
  {
    // The list of events this listener listens to
    def regEx = new RegexEventFilter("xwiki:Code\\..*")
    return Arrays.asList(new DocumentCreatedEvent(regEx), new DocumentUpdatedEvent(regEx))
  }
 // Called by the Observation Manager when an event matches the list of events returned
 // by getEvents()
 void onEvent(Event event, Object source, Object data)
 {
   def context = (XWikiContext) data
   def sender = context.getWiki().getUser(source.getCreator(), context).getEmail()
   def to = "admin@gmail.com"
   def subject = "Document changes on xwiki:Code"
   def body = "Hello, A page has been created or updated in the space 'Code'. You can view this page at: ${source.fullName}"
   def result = context.getWiki().getPluginApi("mailsender", context).sendTextMessage("${sender}", "${to}", subject, body)
  if (source.fullName != "Admin.Logger") {
        def document = context.getWiki().getDocument("Admin.Logger", context)
        document.setContent("${document.getContent()}\n${source.fullName} has been created or updated.")
        context.getWiki().saveDocument(document, "Logging event", true, context)
  }
}
}
// Register against the Observation Manager
def observation = Utils.getComponent(ObservationManager.class)
observation.removeListener("logging")
def listener = new LoggingEventListener()
observation.addListener(listener)
```

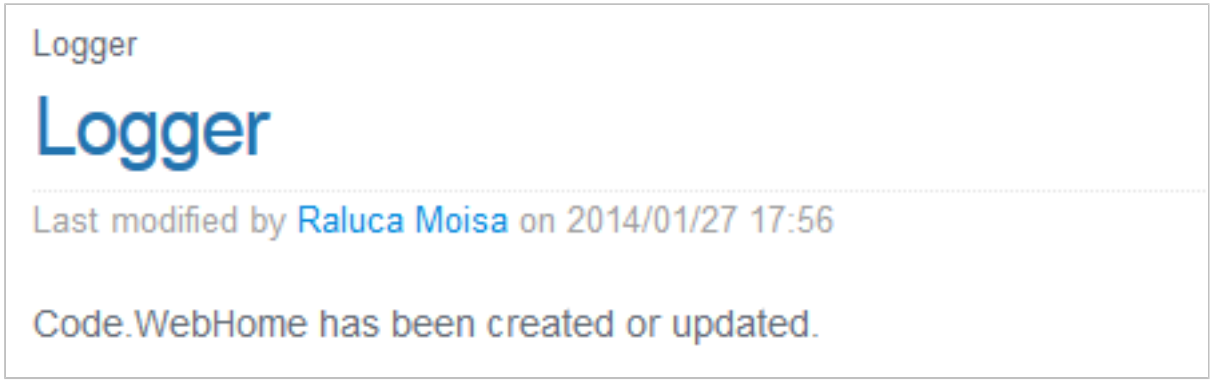
The events the listener listens to are

- DocumentCreatedEvent which means the event is triggered **after** a document is created.
- DocumentUpdatedEvent which means the event is triggered **after** a document is updated.

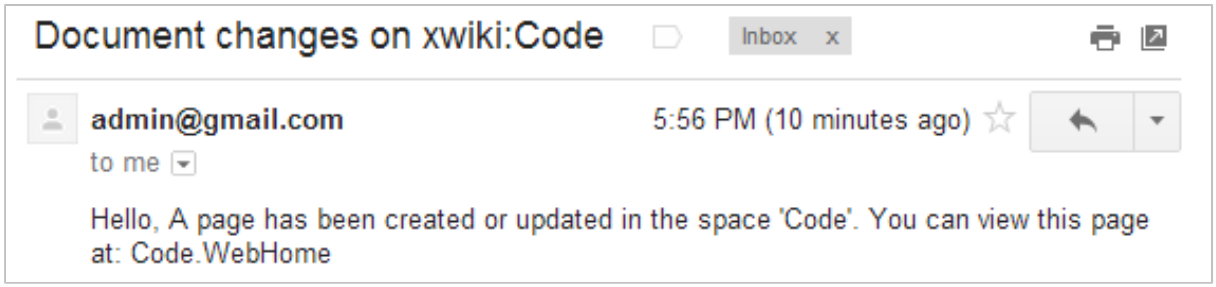
and in order to be able to use them, you must make sure you have imported the dedicated package:

```
import org.xwiki.bridge.event.*
```

After triggering the Scheduler job, create a page in the "xwiki:Code/" space then go to "Admin.Logger" where you should see the new page listed.



Also, provided you have correctly set up a SMTP server, you should receive an email notification regarding the new/updated document.



In case you are interested in more Groovy Event Listener examples, you could check the xwiki.org tutorial.

Related Pages

- **Programming Guide**
  - Programming Overview
  - Post Processing Groovy Listener
  - Groovy Macro

Post Processing Groovy Listener

In the Groovy Notifications tutorial we have presented a way of logging the document create and update events using the XWiki Observation module. However, you might need to make some additional changes on the document after the "Save" action.

A possible use case would be an XWiki application that stores resource names of any kind and for each application entry, the page title is identical to the page name. Supposing, you are already setting the title in the application sheet, you might also need the title to be automatically updated after renaming the entry. Moreover, you might want to avoid manual title changes from other users or even typos. For this purpose, you could create a Scheduler job named "Update Title Listener" with the following code:

```
import org.xwiki.observation.*
import org.xwiki.observation.event.*
import org.xwiki.bridge.event.*
import org.xwiki.observation.event.filter.*
import com.xpn.xwiki.web.*
import com.xpn.xwiki.*
class LoggingEventListener implements EventListener
{
 LoggingEventListener()
 {
 }
 String getName()
 {
   // The unique name of this event listener
   return "loggingTitle"
 }
  List<Event> getEvents()
 {
   // The list of events this listener listens to
  def regEx = new RegexEventFilter("xwiki:RessourcesApplication\\..*")
  return Arrays.asList(new DocumentCreatingEvent(regEx), new DocumentUpdatingEvent(regEx))
 }
// Called by the Observation Manager when an event matches the list of events returned
// by getEvents()
void onEvent(Event event, Object source, Object data)
{
  source.setTitle(source.getName())
}
}
// Register against the Observation Manager
def observation = Utils.getComponent(ObservationManager.class)
observation.removeListener("loggingTitle")
def listener = new LoggingEventListener()
observation.addListener(listener)
```

The events the listener listens to are

- DocumentCreatingEvent which means the event is triggered **before** the document is created.
- DocumentUpdatingEvent which means the event is triggered **before** the document is updated.

and in order to be able to use them, you must make sure you have imported the dedicated package:

```
import org.xwiki.bridge.event.*
```

The complete list of XWiki Bridge Events is available on GitHub at https://github.com/xwiki/xwiki-platform/blob/master/xwiki-platform-core/xwiki-platform-bridge/src/main/java/org/xwiki/bridge/event/. Another tutorial explaining how to update the value of the title field of an XWiki document based on the value of an object's property is available on xwiki.org.

Related Pages

- **Programming Guide**
  - Programming Overview
  - Groovy Notifications

- Groovy Macro

XWiki Platform Plugins

Plugins represented the old way of writing extensions and they were useful when interacting third-party code from the Velocity context. Currently, we are using the XWiki Components.

Architecture

An XWiki plugin consists of

- The plugin itself which implements the XWikiPluginInterface interface. It is also possible to extend the XWikiDefaultPlugin class which is actually an adapter to the "XWikiPluginInterface". The plugin contains the core functionality that are only accessible from scripting provided the user has programming rights.
- The API that extends the Api class and contains all the public methods that are accessible from scripting.

Create and Use a Plugin

As explained in the above section, any plugin has 2 parts: the plugin itself and the API. A simple example of a plugin's core functionality is:

```
package com.xpn.xwiki.plugin.helloworld;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import com.xpn.xwiki.XWikiContext;
import com.xpn.xwiki.api.Api;
import com.xpn.xwiki.plugin.XWikiDefaultPlugin;
import com.xpn.xwiki.plugin.XWikiPluginInterface;
// The plugin class declaration
public class HelloWorldPlugin extends XWikiDefaultPlugin {
 // Create a "log4j" instance in order to log errors
 private static Log LOG = LogFactory.getLog(HelloWorldPlugin.class);
 // Constructor
 public HelloWorldPlugin(String name, String className, XWikiContext context) {
  super(name,className,context);
   init(context);
 }
 // The method used to retrieve the plugin name
 public String getName() {
  return "helloworld";
 }
 // The method used to retrieve the plugin API which requires a cast
 public Api getPluginApi(XWikiPluginInterface plugin, XWikiContext context) {
  return new HelloWorldPluginApi((HelloWorldPlugin) plugin, context);
 }

 // You can additionally overload the cache flush method
 public void flushCache() {
 }
 // Method for initializing the context

 public void init(XWikiContext context) {
    super.init(context);
 }
}
```

Having a log4j instance for the plugin is very useful for debugging. An example of the logger invocation would be:

```
public String getName() {
  LOG.debug("Entered method getName");
 return "helloworld";
}
```

Next, the API code is:

```
package com.xpn.xwiki.plugin.helloworld;
import com.xpn.xwiki.XWikiContext;
import com.xpn.xwiki.api.Api;
// The class declaration
public class HelloWorldPluginApi extends Api {

  //The plugin field declaration that will allow the API to call backend methods.
  private HelloWorldPlugin plugin;
  //The constructor
  public HelloWorldPluginApi(HelloWorldPlugin plugin, XWikiContext context) {
     super(context);
       setPlugin(plugin);
  }
  //The plugin getter

  public HelloWorldPlugin getPlugin(){
     return (hasProgrammingRights() ? plugin : null);
     // Uncomment for allowing unrestricted access to the plugin
     // return plugin;
  }
  //The plugin setter
  public void setPlugin(HelloWorldPlugin plugin) {
     this.plugin = plugin;
  }
  //The API method that can be called from Velocity
  public String hello() {
     return "Hello World!";
  }
  public void updatePage() {
     //...
  }
}
```

In order to use the plugin, copy your classes to the XWiki "webapps/xwiki/WEB-INF/classes" folder, or copy the JAR file containing the classes in the "webapps/xwiki/WEB-INF/lib" folder. Finally, you need to register the new plugin in the "WEB-INF/xwiki.cfg" file by adding the line

```
...,\
com.xpn.xwiki.plugin.helloworld.HelloWorldPlugin
```

then restart the server.

To use the plugin in a wiki page, just write

```
"$xwiki.helloworld.hello()"
```

which will display "Hello World!".

XWiki Platform Plugins

Plugins that are in their own JAR

- Activity Stream Plugin
- Application Manager Plugin was written to work with the Application Manager Application.
- Calendar Plugin
- eXo Platform
- Chart Plugin
- Joda Time Plugin
- Lucene Plugin
- Mail Sender Plugin
- Scheduler Plugin
- Skin Extension Plugin
- Swizzle Plugin allows you to query a JIRA instance in order to display related information in a XWiki page. We recommend using the JIRA Module but note that the JIRA REST API is still young and doesn't yet offer all the possibilities of the older "XMLRPC/SOAP API" used by Swizzle.

- Tag Plugin
- Watch List Plugin
- Wiki Manager Plugin depends on the Application Manager Plugin and it was written to work with the Wiki Manager Application.
- Zip Explorer Plugin

Plugins that are in the XWiki Old Core

- The old diff Plugin which is deprecated since XWiki 4.1. We now use the diff service. This plugin returns a list of Delta objects representing line differences in text1 and text2.
- The file upload Plugin offers access to uploaded files. The uploaded files are automatically parsed and preserved as a list of `{@link org.apache.commons.fileupload.FileItem}s`.
- The graphviz plugin wraps the GraphVi executable transforming dot source files (representing graphs) into images, image maps, or other output formats supported by GraphViz.
- image Plugin
- ldap Plugin
- mail Plugin
- packaging Plugin
- rightsmanager Plugin
- svg Plugin

Related Pages

- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki API Reference
  - The Watchlist Plugin
  - Tag Plugin
  - Skin Extension Plugin
  - Scheduler Plugin
  - Programming Overview
  - Mail Sender Plugin
  - Lucene Plugin
  - Joda Time Plugin
  - Charting Plugin
- **Admin Guide**
  - Other XWiki Applications

Activity Stream Plugin

The Activity Stream plugin provides APIs to store wiki events in the `activitystream_events` table. This way the runtime processing of heavy XWiki documents is avoided. Events can be filtered by:

- wiki or space
- the name of the activity stream they have been recorded to
- the user that has triggered the record

To learn more about these parameters and how to customize the Activity Stream, check the documentation page about the Activity Stream macro. Also, the source code is available on GitHub at xwiki-platform-activitystream.

The Activity Stream plugin has a dedicated section in the `xwiki/WEB-INF/xwiki.cfg` configuration file which allows you to decide whether to store the events for each wiki in its own database or into the main wiki database. By default, both options are enabled.

You can also set the number of days to keep the events in the database. The default value is `0` which means the events are kept forever.

```
#-# [Since 2.0RC1]
#-# Activity Stream plugin.
#-# The Activity Stream plugin stores data in a dedicated table in the database. Each wiki has its own database.
#-# The plugin can be configured to store its data into the database corresponding to the wiki, into the main database
#-# (default: xwiki) or both. These options should not be both set to 0 (in this case the local store will be forced).
#-# Important note: disabling storage in the main store will prevent the watchlist from retrieving events from subwikis.
#-#
#-# Default: 1
# xwiki.plugin.activitystream.uselocalstore=1
#-#
#-# Default: 1
# xwiki.plugin.activitystream.usemainstore=1
#-#
#-# Number of days the events should be kept (0 or any negative value: infinite duration)
#-# Note: if this value is greater than 0 a scheduler job will be created, this job will then be fired every week to
#-# delete events older than the configured value.
#-# Default: 0
# xwiki.plugin.activitystream.daystokeepevents=0
```

The Activity Stream plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.activitystream.plugin.ActivityStreamPlugin, \
...
```

The custom Hibernate mapping is also activated by default in the `WEB-INF/hibernate.cfg.xml` file, under your database standard XWiki configuration. If you are using MySQL, the corresponding line is:

```
<mapping resource="activitystream.hbm.xml"/>
```

Related Pages

- **Programming Guide**
  - Activity Macro

Charting Plugin

The Charting Plugin allows drawing charts using JFreeChart. The plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.charts.ChartingPlugin,\
...
```

The source code is available on GitHub at xwiki-platform-chart.

Given that for the moment XWiki Enterprise does not provide a dedicated wizard, you can use the Chart Macro to display charts from given data sources.

Related Pages

- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki Platform Plugins
  - XWiki API Reference
  - The Watchlist Plugin
  - Tag Plugin
  - Skin Extension Plugin
  - Scheduler Plugin
  - Programming Overview
  - Mail Sender Plugin
  - Lucene Plugin
  - Joda Time Plugin
  - Chart Macro

Joda Time Plugin

The Joda Time plugin provides a Velocity-friendly API on top of the Joda Time library. The plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
#-# List of active plugins.
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.jodatime.JodaTimePlugin,\
...
```

The source code is available on GitHub at xwiki-platform-jodatime.

Examples

```
{{velocity}}
## Format the displayed result
#set($formatter = $xwiki.jodatime.getDateTimeFormatterForStyle('F-'))
## Display the current time
#set($currentTime = $xwiki.jodatime.dateTime)
$formatter.print($currentTime)
## Display the second day of the current week
#set($secondDayWeek = $currentTime.withDayOfWeek(2))
$formatter.print($secondDayWeek)
{{/velocity}}
```

will display

Example using "getDateTimeFormatterForPattern"

```
{{velocity}}
#set($formatter = $xwiki.jodatime.getDateTimeFormatterForPattern("dd/MM/yyyy 'at' hh:mm"))
## Display the document creation time
#set($creationDate = $xwiki.jodatime.getDateTime($doc.creationDate.time))
Document created on $formatter.print($creationDate)
## Display the number of days left until Easter
#set($formatter = $xwiki.jodatime.getDateTimeFormatterForPattern('yyyy.MM.dd'))
#set($easterDay = $formatter.parseDateTime('2014.04.20'))
#set($now = $xwiki.jodatime.dateTime)
#set($daysLeft = $easterDay.minus($now.millis))
$daysLeft.getDayOfYear() days left until Easter
{{/velocity}}
```

will display

More examples for the Joda Time plugin are available on xwiki.org.

Related Pages

- **Programming Guide**
  - Zip Explorer Plugin

- XWiki Platform Plugins
- XWiki API Reference
- The Watchlist Plugin
- Tag Plugin
- Skin Extension Plugin
- Scheduler Plugin
- Programming Overview
- Mail Sender Plugin
- Lucene Plugin
- Charting Plugin

Lucene Plugin

The Lucene Plugin uses Lucene to asynchronously index the documents in the wiki/farm. The plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.lucene.LucenePlugin,\
...
```

The dedicated section in the `xwiki.cfg` file allows you to further customize the Lucene search engine:

```
#-# Lucene search engine
#-# Location where to place the lucene index files. The default is the "lucene" subdirectory in XWiki's permanent data
#-# directory. Change it if you want to store indexes in another place.
# xwiki.plugins.lucene.indexdir=/usr/local/xwiki/lucene
#-# The text analyzer to use for indexing.
# xwiki.plugins.lucene.analyzer=org.apache.lucene.analysis.standard.StandardAnalyzer
#-# The number of seconds to wait between reindexes. A smaller value ensures that new documents will be indexed faster,
#-# but with a minor performance reduction. Adjust according to your wiki load.
# xwiki.plugins.lucene.indexinterval=20
#-# The maximum size of the indexing queue. After this limit is reached, the reindex thread will have to wait until the
#-# queue is consumed. Note that this does not affect documents submitted through the notification mechanism, only the
#-# full reindex option.
# xwiki.plugins.lucene.maxQueueSize=1000
```

The default folder that stores the Lucene index is `<permanent directory>/lucene` but you can change it by editing the `xwiki.plugins.lucene.indexdir` parameter.

The `xwiki.plugins.lucene.analyzer` parameter is the default Lucene Analyzer which converts all words in lowercase and filters out simple words such as "the", "a". Other available analyzers are listed at this link.

The `xwiki.plugins.lucene.indexinterval` parameter controls how frequently the plugin starts an indexation thread. The value is in seconds.

The `xwiki.plugins.lucene.maxQueueSize` parameter sets the maximum number of elements in the indexing queue. Most importantly, this property only affects the "Index the whole farm" option.

More information is available in the "Search Application" and "Customizing the Search Feature" documentation pages listed in the "Related Pages" section.

The source code is available on GitHub at xwiki-platform-search-lucene.

Related Pages

- **User Guide**
  - Search Application
- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki Platform Plugins
  - XWiki API Reference
  - The Watchlist Plugin
  - Tag Plugin
  - Skin Extension Plugin
  - Scheduler Plugin
  - Programming Overview
  - Mail Sender Plugin
  - Joda Time Plugin
  - Charting Plugin
- **Admin Guide**
  - Search Suggest Sources
  - Customize the Search Feature

Mail Sender Plugin

The Mail Sender Plugin provides an API for sending emails and it is used by:

- the Watchlist plugin
- the Send page by email feature
- the Invitation Application

The plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.mailsender.MailSenderPlugin,\
...
```

The source code is available on GitHub at xwiki-platform-mailsender. As you can see in the API, the key methods are:

```
* public int sendHtmlMessage(String from, String to, String cc, String bcc, String subject, String body,
      String alternative, List<Attachment> attachments) {}
* public int sendTextMessage(String from, String to, String subject, String message){}
* public int sendTextMessage(String from, String to, String cc, String bcc, String subject, String message,
      List<Attachment> attachments) {}
* public int sendMessageFromTemplate(String from, String to, String cc, String bcc, String language,
      String documentFullName, VelocityContext vcontext) {}
* public int sendMessageFromTemplate(String from, String to, String cc, String bcc, String language,
      String documentFullName, Map<String, Object> parameters) {}
```

where:

- @*param from* is the e-mail sender
- @*param to* is the e-mail recipient
- @*param cc* is the e-mail Carbon Copy
- @*param bcc* is the e-mail Hidden Carbon Copy
- @*param subject* is the subject of the message
- @*param body*, @*param message* is the content of the message
- @*param language* is the language of the e-mail
- @*param alternative* is the alternative text offered to the mail client
- @*param attachments* is a list of "com.xpn.xwiki.api.Attachment" that will be attached to the mail
- @*param documentFullName* is the full name of the template to be used. The template must have an "XWiki.Email" object attached
- @*param parameters* are variables to be passed to the velocity context

An example using the Mail Sender plugin in a Scheduler job is available in the Groovy Notification Tutorial.

Related Pages

- **User Guide**
  - XWiki Watchlist Application
  - Watch a Wiki
  - Watch a Space
  - Watch a Page
  - Share Page by Email
  - More Page Actions
  - Activity Stream
- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki Platform Plugins
  - XWiki API Reference
  - The Watchlist Plugin
  - Tag Plugin
  - Skin Extension Plugin
  - Scheduler Plugin
  - Programming Overview
  - Lucene Plugin

- Joda Time Plugin
- Charting Plugin

Scheduler Plugin

The Scheduler Plugin provides an API for scheduling jobs that are executed periodically or a single time. Also, jobs can be written as Groovy scripts or as Java classes. The plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.scheduler.SchedulerPlugin,\
...
```

The source code is available on GitHub at xwiki-platform-scheduler.

To learn how to create Scheduler jobs, you could also check:

- the Scheduler Application documentation page
- the Groovy Notification Tutorial
- the Post Processing Groovy Listener Tutorial

Related Pages

- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki Platform Plugins
  - XWiki API Reference
  - The Watchlist Plugin
  - Tag Plugin
  - Skin Extension Plugin
  - Programming Overview
  - Mail Sender Plugin
  - Lucene Plugin
  - Joda Time Plugin
  - Charting Plugin

Skin Extension Plugin

The Skin Extension Plugin allows interface components to pull CSS and JavaScript resources in order to avoid the inclusion by default of styling and scripting files that are only used in some pages. The Skin Extension Plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.skinx.JsSkinExtensionPlugin,\
  com.xpn.xwiki.plugin.skinx.JsSkinFileExtensionPlugin,\
  com.xpn.xwiki.plugin.skinx.JsResourceSkinExtensionPlugin,\
  com.xpn.xwiki.plugin.skinx.CssSkinExtensionPlugin,\
  com.xpn.xwiki.plugin.skinx.CssSkinFileExtensionPlugin,\
  com.xpn.xwiki.plugin.skinx.CssResourceSkinExtensionPlugin,\
  com.xpn.xwiki.plugin.skinx.LinkExtensionPlugin,\
...
```

The source code is available on GitHub at xwiki-platform-skin.

Usage

- `$xwiki.jsx.use('PageFullName')` pulls a JavaScript skin extension located in the wiki.
- `$xwiki.ssx.use('PageFullName')` pulls a style sheet skin extension located in the wiki.

- `$xwiki.jsfx.use('path/to/file.js')` pulls a JavaScript file located in the current skin folder or in the "resources" folder.
- `$xwiki.ssfx.use('path/to/file.css')` pulls a style sheet file located in the current skin folder or in the resources folder.

- `$xwiki.jsrx.use('path/to/file.js')` pulls a JavaScript file located in a JAR located either in the "WEB-INF/lib" folder or added through an extension.
- `$xwiki.ssrx.use('path/to/file.css')` pulls a style sheet file located in a JAR located either in the "WEB-INF/lib" folder or added through an extension.

Extra Parameters

In all the above use cases is possible to have a map of parameters as a second parameter. For instance:

```
$xwiki.jsx.use('Document.Name', {'minify' : false, 'language': $context.language)
```

The `minify` parameter prevents minifying the JavaScript and CSS code for debug purposes. The default value is `true`.

The `language` parameter is used by default for JavaScript extensions and it is set to the current context language, so that messages can be correctly translated with `$services.localization.render`.

In case you are using `jsfx` or `ssfx` there is an optional second Boolean parameter which allows you to specify whether the URL of the pulled file extension should always pass through the skin action. For instance:

```
$xwiki.jsfx.use('path/to/file.js', true);
```

To enable this behavior with the generic parameter map, you can use the syntax below:

```
$xwiki.jsfx.use('file.js', {'forceSkinAction': true})
```

The `forceSkinAction` parameter is used only for skin resources and it forces to construct URLs which pass through the skin action, which enables the Velocity parsing of the file. The default value is `false`.

Other 2 additional parameters are:

- `defer` is used only for JavaScript extensions to prevent scripts from being deferred. The default value is `true`.
- `colorTheme` allows to render a StyleSheet extension using a specific color theme, including the ones from other wikis.

To learn how to create XWiki "JavaScript" and "StyleSheet" extensions, check the dedicated documentation page.

Related Pages

- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki Platform Plugins
  - XWiki API Reference
  - The Watchlist Plugin
  - Tag Plugin
  - Skins Extensions
  - Scheduler Plugin
  - Programming Overview
  - Mail Sender Plugin
  - Lucene Plugin
  - Joda Time Plugin
  - Create and Override a Skin
  - Charting Plugin
- **Admin Guide**
  - Color Theme Application

Tag Plugin

The Tag Plugin provides the API for creating, renaming and deleting tags. The Tag Plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.tag.TagPlugin, \
...
```

The source code is available on GitHub at xwiki-platform-tag.

To learn more about the XWiki tags, you might also be interested in:

- the XWiki Tag Application documentation page listed in the "Related Pages" section
- the Tag Cloud Macro documentation page

Related Pages

- **User Guide**
  - Tags Application
- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki Platform Plugins
  - XWiki API Reference
  - The Watchlist Plugin
  - Tag Cloud Macro
  - Skin Extension Plugin
  - Scheduler Plugin
  - Programming Overview
  - Mail Sender Plugin
  - Lucene Plugin
  - Joda Time Plugin
  - Charting Plugin

The Watchlist Plugin

The Watchlist Plugin uses the Mail Sender Plugin and allows users to create a list of wikis, spaces and pages for which to monitor the activity and receive notification emails. The plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.watchlist.WatchListPlugin, \
...
```

The source code is available on GitHub at xwiki-platform-watchlist.

Other useful documentation pages are listed in the "Related Pages" section.

Related Pages

- **User Guide**
  - XWiki Watchlist Application
  - Watch a Wiki
  - Watch a Space
  - Watch a Page
  - More Page Actions
  - Activity Stream
- **Programming Guide**
  - Zip Explorer Plugin
  - XWiki Platform Plugins
  - XWiki API Reference
  - Tag Plugin
  - Skin Extension Plugin
  - Scheduler Plugin
  - Programming Overview
  - Mail Sender Plugin
  - Lucene Plugin
  - Joda Time Plugin
  - Charting Plugin

Zip Explorer Plugin

The Zip Explorer Plugin provides an API to create links to files inside a ZIP file. The plugin is enabled by default in XWiki Enterprise, so you will find it in the "xwiki.plugins" list of the `xwiki.cfg` file:

```
xwiki.plugins=\
...
  com.xpn.xwiki.plugin.zipexplorer.ZipExplorerPlugin,\
...
```

The source code is available on GitHub at xwiki-platform-zipexplorer.

The Zip Explorer API lists the content of ZIP attachments (including XAR files) and listens to the XWiki `download` action in order to display the attachment content. The accepted URLs have the following syntax:

```
http://[...]/download/WikiPageName/zipfile.zip/FolderName/FileName.txt
```

Examples

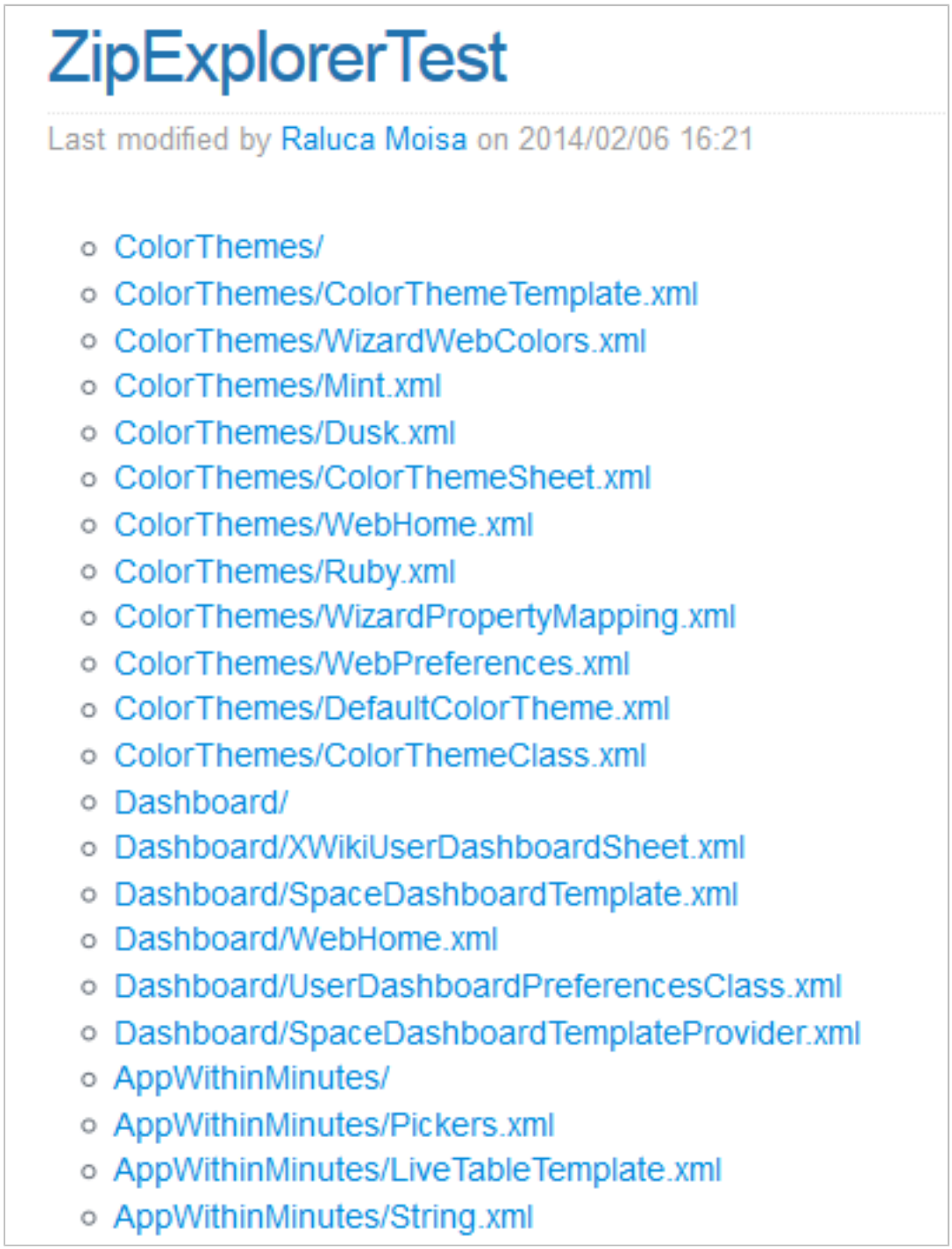
Supposing you have an attachment named "xwiki-commons-component-api-5.3-javadoc.jar" which contains a Javadoc and you want to create a link to the "index.html" file, the "xwiki/2.1" syntax will be:

```
{{velocity}}
[[XWiki Commons - Component - API 5.3 API Javadoc>>path:$xwiki.zipexplorer.getFileLink($doc, "xwiki-commons-component-api-5.3-javadoc.jar", "index.html")]]
{{/velocity}}
```

To list the content of the "xwiki-enterprise-ui-all-4.5.3.xar" file attached to the current page, use the snippet below:

```
{{velocity}}
#set($myXAR="xwiki-enterprise-ui-all-4.5.3.xar")
#set($plugin=$xwiki.zipexplorer)
#foreach($file in $plugin.getFileTreeList($doc, $myXAR))
 * [[$file.id>>path:$plugin.getFileLink($doc, $myXAR, $file.id)]]
#end
{{/velocity}}
```

The output will then be:



Related Pages

- **User Guide**
  - Page Attachments
- **Programming Guide**
  - XWiki Platform Plugins
  - XWiki API Reference
  - The Watchlist Plugin
  - Tag Plugin
  - Skin Extension Plugin
  - Scheduler Plugin
  - Programming Overview

    - Mail Sender Plugin
    - Lucene Plugin
    - Joda Time Plugin
    - Charting Plugin
    - Attachment Selector Macro
- **Admin Guide**
    - WebDAV
    - Attach Large Files