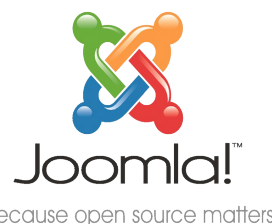


Available extension parameter types and how to use them

When you create an extension (component, module, plugin or template) you can implement parameters in the .XML file. With this parameters you can set options, and use them in the main file, the .PHP file.

There are 20 extension parameter types available in Joomla! 1.5. You can use them easily and in this document you will find a description for all of them. You can also add your own parameters, but this will not be described in this document.



License

This document is released under [Joomla! Electronic Documentation License](#).

Author

This document is made by [Marieke van der Tuin](#).

Table of contents

- XML file
- Parameters
 - Core parameters
 - Calendar
 - Category
 - Editors
 - File list
 - Folder list
 - Help sites
 - Hidden
 - Image list
 - Language
 - List
 - Menu
 - Menu item
 - Password
 - Radio
 - Section
 - Spacer
 - SQL
 - Text
 - Text area
 - Time zones

XML file

You need an XML file for every extension. It is used mainly to install the extension. A particular XML file looks like this:

```
<?xml version="1.0" encoding="utf-8"?>
<install version="1.5" type="extension type">
  <name>Name of your extension</name>
  <creationDate>Created Date</creationDate>
  <author>Your name</author>
  <authorEmail>Your e-mail address</authorEmail>
  <authorUrl>Your website</authorUrl>
  <copyright>Copyright</copyright>
  <license>License, for example GNU/GPL</license>
  <version>Version of the extension</version>
  <description>Description of the extension</description>
  <files>
    <filename>add the files between those tags</filename>
  </files>
  <languages>
    <language tag="en-GB">Language file</language>
  </languages>
  <params>
    Place the parameters between these tags.
  </params>
</install>
```

Parameters

The parameters should be placed between the `<params>` and `</params>` tags. You can also add groups. For example, the 'Advanced' group. This will look like this:

```
<params>
  'Normal' parameters
</params>
<params group="advanced">
  Advanced parameters
</params>
```

Note: You can not add groups to templates.

Core Parameters

There are 20 parameter types available within your Joomla! 1.5 installation. These are: Calendar, Category, Editors, File list, Folder list, Help sites, Hidden, Image list, Language, List, Menu, Menu item, Password, Radio, Section, Spacer, SQL, Text, Text area and Time zones, and will be described here in detail.

Each one is described in the following order:

- Description
- Screen shot of output
- XML file
- Implementation in PHP file
- Reference

Calendar

Description: This parameter shows a text box where you can fill in the date. You can also choose the date from a calendar, which pops up after you clicked on the icon next to the text box.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="calendar" type="Calendar" default="5-10-2008"
label="Calendar" description="" format="%d-%m-%Y" />
```

Name: The name used to implement in the PHP file.
Type: For a calendar parameter, use 'Calendar'.
Default: The default date.
Label: The name displayed at the output of the parameter.
Description: The description displayed as a tool tip.
Format: The format of the date.

PHP file: For example, get the parameter, named `publish_up`, like this:

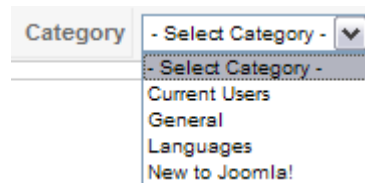
```
$publish_up = new JDate($row->publish_up);
```

Reference: administrator\components\com_content\admin.content.html.php, 122-168

Category

Description: This parameter shows a drop down list of categories from a section.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="category" type="Category"
label="Category" description="" section="3" />
```

Name: The name used to implement in the PHP file.
Type: For a category parameter, use 'Category'.
Label: The name displayed at the output of the parameter.
Description: The description displayed as a tool tip.
Section: The section ID number, can be found in the Section Manager.

PHP file: For example, get use the parameter, named `category`, like this:

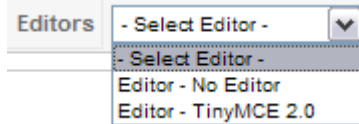
```
$category[$section->id][ ]
```

Reference: administrator\components\com_content\controller.php, 452-530

Editors

Description: This parameter shows a drop down list of the available WYSIWYG editors.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="editors" type="Editors" default=""  
label="Editors" description="" />
```

Name: The name used to implement in the PHP file.

Type: For an editor parameter, use 'Editor'.

Default: The default editor.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tool tip.

PHP file: For example, get the parameter, named editors, like this:

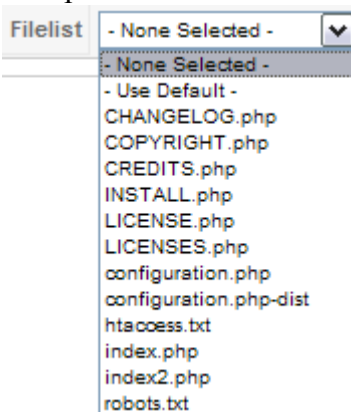
```
$this->lists['Editors']
```

Reference: administrator\components\com_config\controllers\application.php, 86

File list

Description: This parameter shows a drop down list of files from a certain directory.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="filelist" type="Filelist" default=""  
label="File list" description="" directory=""  
filter="" exclude="" striptext="" />
```

Name: The name used to implement in the PHP file.

Type: For a file list parameter, use 'Filelist'.

Default: The default file.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tool tip.

Directory: The directory from which the files should be showed.
For example "administrator".

Filter: Search in the files and only show the files containing this word or letter.

Exclude: Exclude a certain file, file format or word from showing in the list.

Stripext: Strip these characters.

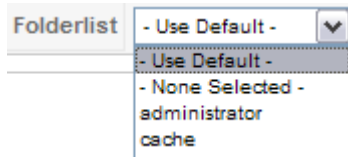
PHP file: For example, get the parameter, named editors, like this:

```
$params->get('ParameterName')
```

Folder list

Description: This parameter shows a drop down list of folders from a certain directory.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="folderlist" type="Folderlist" default=""  
label="Folder list" description="" directory=""  
filter="" exclude="" striptext="" />
```

Name: The name used to implement in the PHP file.

Type: For a folder list parameter, use 'Folderlist'.

Default: The default folder.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tooltip.

Directory: The directory from which the folders should be showed.
For example "administrator".

Filter: Search within the folders and only show the folders containing this
word or letter.

Exclude: Exclude a certain folder or some folders containing a certain word
from showing in the list.

Stripext: Strip these characters.

PHP file: For example, get the parameter, named headerColor, like this:

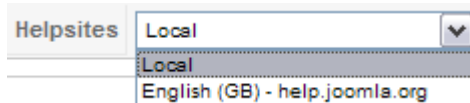
```
$this->params->get('headerColor', 'green')
```

Reference: administrator\templates\khepri\cpanel.php, 45

Help sites

Description: This parameter shows a drop down list of the help sites of your Joomla! Installation.
These can be found at administrator\help\helpsites-15.xml

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="helpsites" type="Helpsites"  
default="local" label="Help sites" description="" />
```

Name: The name used to implement in the PHP file.

Type: For a help site parameter, use 'Helpsites'.

Default: The default editor.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tooltip.

PHP file: For example, get the parameter, named helpsites, like this:

```
$this->params->get('helpsites', 'local')
```

Reference: administrator\components\com_config\controllers\application.php, 90-94

Hidden

Description: This parameter collects information about the user viewing the parameter section.

Screen shot: As the name says; this parameter is a hidden one. No output will be displayed.

XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="hidden" type="Hidden" value="" />
```

Name: The name used to implement in the PHP file.

Type: For a hidden parameter, use 'Hidden'.

Value: The data which needs to be collected.

Two examples:

Get ID of user:

```
value="<?php echo $this->user->get('id'); ?>"
```

Get filter order:

```
value="<?php echo $this->lists['order']; ?>"
```

PHP file: For example, get the parameter, named id, like this:

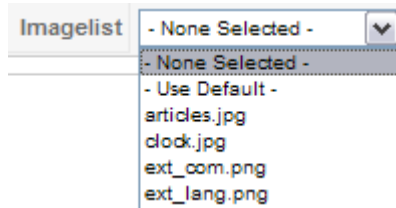
```
$user->get('id')
```

Reference: administrator\components\com_users\views\user\view.html.php, 52-76

Image list

Description: This parameter shows a drop down list with images; .png, .gif, .jpg, .bmp and .ico.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="imagelist" type="Imagelist" default=""  
label="Image list" description="" directory="" exclude=""  
stripext="_" />
```

Name: The name used to implement in the PHP file.

Type: For an image list parameter, use 'Imagelist'.

Default: The default image.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tooltip.

Directory: The directory from which the images should be showed.
For example "images/stories".

Exclude: Exclude a certain image name, format or word from showing in the list.

Stripext: Strip these characters.

PHP file: For example, get the parameter, named image, like this:

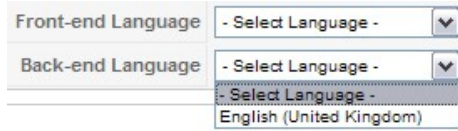
```
JHTML::_('list.images', 'image', $row->image );
```

Reference: administrator\components\com_categories\admin.categories.php, 386

Language

Description: This parameter shows a drop down list with the installed languages for Front-end or Back-end.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this

```
<param name="language" type="Languages" client="site"
default="" label="Front-end Language" description="" />
```

Name: The name used to implement in the PHP file.

Type: For a language parameter, use 'Languages'.

Client: Use 'site' when you want to display the Front-end languages, use 'administrator' when you want to display the Back-end languages.

Default: The default language.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tooltip.

PHP file: For example, get the parameters like this:

```
$params = $this->user->getParameters(true);
echo $params->render( 'params' );
```

Reference: administrator/components/com_users/views/user/tmpl/form.php, 188-189

List

Description: This parameter shows a drop down list with options.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="list" type="List" default="" label="List"
description="" >
    <option value="0">Label option 1</option>
    <option value="1">Label option 2</option>
</param>
```

Name: The name used to implement in the PHP file.

Type: For a list parameter, use 'List'.

Default: The default list option.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tooltip.

Option: The option. Place the label between <option> and </option>. You can add as many as you would like.

Option value: The name used to implement in the PHP file.

Don not forget to close the parameter with </param>

PHP file: For example, get the parameter, named colorVariation, like this:

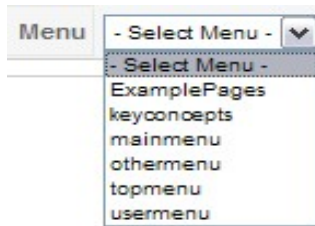
```
$this->params->get('colorVariation')
```

Reference: templates/rhuk_milkyway/index.php,

Menu

Description: This parameter shows a drop down list with the menus from your Joomla! site.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="menu" type="Menu" default="" label="Menu"
description="" />
```

Name: The name used to implement in the PHP file.

Type: For a menu parameter, use 'Menu'.

Default: The default menu.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tooltip.

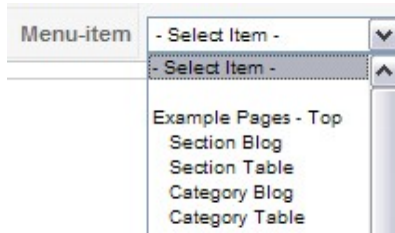
PHP file: For example, get the parameter, named menu, like this:

```
$this->params->get('menu')
```

Menu item

Description: This parameter shows a drop down list with the menu-items from your Joomla! site.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="menuitem" type="MenuItem" default=""
label="Menu-item" description="" />
```

Name: The name used to implement in the PHP file.

Type: For a menu parameter, use 'MenuItem'.

Default: The default menu-item.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tooltip.

PHP file: For example, get the parameter, named selections, like this:

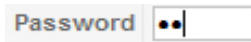
```
$lists['selections']
```

Reference: administrator\components\com_templates\admin.templates.html.php, lines 325 – 355 (create JavaScript) and 382

Password

Description: This parameter shows a text box, where you can fill in a (hidden) password.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="password" type="Password" default=""  
label="Password" description="" size="5" />
```

Name: The name used to implement in the PHP file.
Type: For a password parameter, use 'Password'.
Default: The default password.
Label: The name displayed at the output of the parameter.
Description: The description displayed as a tooltip.
Size: The width of the text box.

PHP file: For example, get the parameter, named password, like this:

```
$this->user->get('password')
```

Reference: administrator\components\com_users\views\user\tmpl\form.php, 125

Radio

Description: This parameter shows radio buttons to select different options.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="radio" type="Radio" default="0" label="Radio"  
description="" >  
    <option value="0">1</option>  
    <option value="1">2</option>  
</param>
```

Name: The name used to implement in the PHP file.
Type: For a radio parameter, use 'Radio'.
Default: The default radio option.
Label: The name displayed at the output of the parameter.
Description: The description displayed as a tooltip.
Option: The option. Place the label between <option> and </option>. You can add as many as you would like.

Option value: The name used to implement in the PHP file.

Don not forget to close the parameter with </param>

PHP file: For example, get the parameter, named smtpauth, like this:

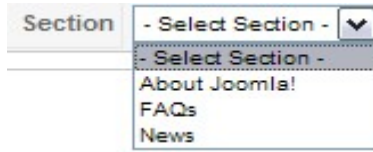
```
$lists['smtpauth'] = JHTML::_('select.booleanlist',  
'smtpauth', 'class="inputbox"', $row->smtpauth);
```

Reference: administrator\components\com_config\controllers\application.php, 154

Section

Description: This parameter shows a drop down list with the sections from your Joomla! site.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="section" type="Section" default="" label="Section"
description="" />
```

Name: The name used to implement in the PHP file.

Type: For a section parameter, use 'Section'.

Default: The default section.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tooltip.

PHP file: For example, get the parameter, named sectionid, like this:

```
$lists['sectionid'] = JHTML::_('select.genericlist', $sections,
'sectionid', 'class="inputbox" size="1" '.$javascript, 'id',
'title', intval($row->sectionid));
```

Reference: administrator\components\com_content\controller.php, 462

Spacer

Description: This parameter shows a separator between the other parameters.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param type="Spacer" />
```

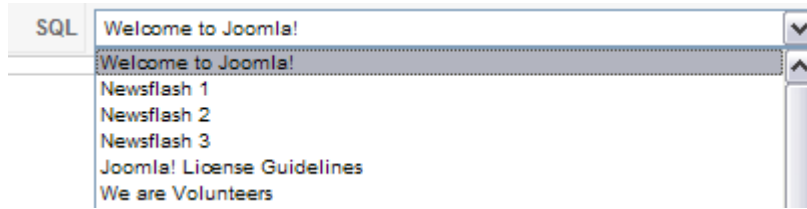
Type: For a spacer parameter, use 'Spacer'.

PHP file: You can not implement this in your PHP file, it is just a separator, a single line.

SQL

Description: This parameter creates a drop down list based on the database.

Screen shot:



XML file: Use the following code in the XML file to create a parameter like this.

```
<param name="title" type="SQL" default="" label="SQL"
description="" query="SELECT id, title FROM #__content" />
```

Name: The name used to implement in the PHP file. This should be the same as the label of the query.

Type: For a sql parameter, use 'SQL'.

Default: The default item, use the value from the query to define it.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tooltip.

Query: Use any query to the database you want. For example, use 'SELECT':

```
query="SELECT value, label FROM #__table"
```

Replace the 'value' and 'label' by existing columns of your database. Replace 'table' by a table from the database. Do not add the prefix. For example, jos_categories will become #__categories.

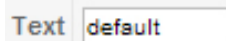
PHP file: For example, get the parameter, named title, like this:

```
$this->params->get('title')
```

Text

Description: This parameter shows a text box.

Screen shot:



XML file:

```
<param name="text" type="Text" default="default" label="Text"
description="" size="10" />
```

Name: The name used to implement in the PHP file.

Type: For a text parameter, use 'Text'.

Default: The default text.

Label: The name displayed at the output of the parameter.

Description: The description displayed as a tool tip

Size: The width of the text box.

PHP file: For example, get the parameter, named count, like this:

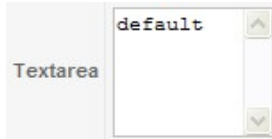
```
$count = intval($params->get('count', 20));
```

Reference: modules\mod_sections\helper.php, 28

Text area

Description: This parameter shows a text area.

Screen shot:



XML file:

```
<param name="textarea" type="Textarea" default="default"
label="Text area" description="" rows="10" cols="5" />
```

Name: The name used to implement in the PHP file.
Type: For a text area parameter, use 'Textarea'.
Default: The default text.
Label: The name displayed at the output of the parameter.
Description: The description displayed as a tool tip
Rows: The number of rows of the text area.
Cols: The number of columns of the text area.

PHP file: For example, get the parameter, named area, like this:

```
$this->params->get('title')
```

Time zones

Description: This parameter shows a drop down list with all time zones.

Screen shot:



XML file:

```
<param name="timezones" type="Timezones" default="-10"
label="Timezones" description="" />
```

Name: The name used to implement in the PHP file.
Type: For a time zone parameter, use 'Timezones'.
Default: The default time zone. Use for example '-10' for UTC -10:00
Label: The name displayed at the output of the parameter.
Description: The description displayed as a tool tip

PHP file: For example, get this and other parameters, like this:

```
$this->params->render( 'params' )
```

Reference: components\com_user\views\user\tmpl\form.php, 86