# DW Anything Studio v1.02 User Manual

Ilya Vikharev

# DW Anything Studio v1.02 User Manual

Ilya Vikharev

Publication date 2017

# Table of Contents

# List of Tables

# Chapter 1. Introduction to DW Anything Studio

DW Anything Studio is a component for Joomla! CMS designed to build custom dynamic pages to display any data from your site's database.

With DW Anything Studio, you define page layout templates and datasets that retrieve data from database and supply it to your templates. Thus you can build complex dynamic pages, data reports, and custom applications without any programming.

DW Anything Studio features include:

- Easy to use yet sophisticated templating system using the power of a popular Smarty templating engine
- Full access to your site's database using MySQL queries
- Support for unlimited levels query nesting

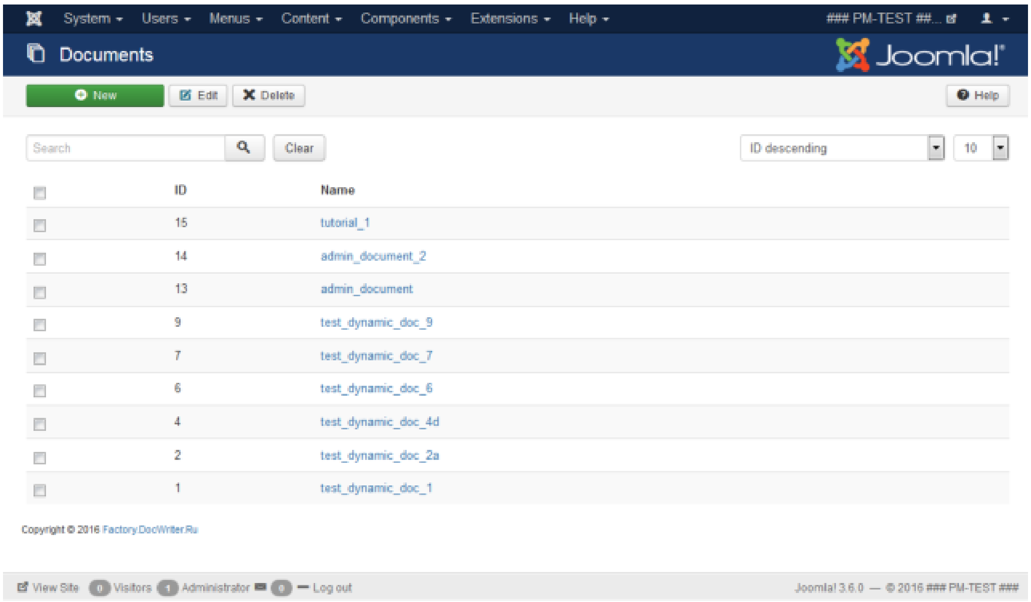
# Chapter 2. Documents

Documents are the most basic entities in DW Anything Studio.

Each document is used to establish a connection between a *template* and a *dataset*. It defines which information will be taken from database and which template will be used to represent this information on your Joomla!-driven site.

## 1. Managing Documents List

The documents list displays all the DW Anything Studio document definitions.



| □ | ID | Name |
|---|---|---|
| □ | 15 | tutorial_1 |
| □ | 14 | admin_document_2 |
| □ | 13 | admin_document |
| □ | 9 | test_dynamic_doc_9 |
| □ | 7 | test_dynamic_doc_7 |
| □ | 6 | test_dynamic_doc_6 |
| □ | 4 | test_dynamic_doc_4d |
| □ | 2 | test_dynamic_doc_2a |
| □ | 1 | test_dynamic_doc_1 |

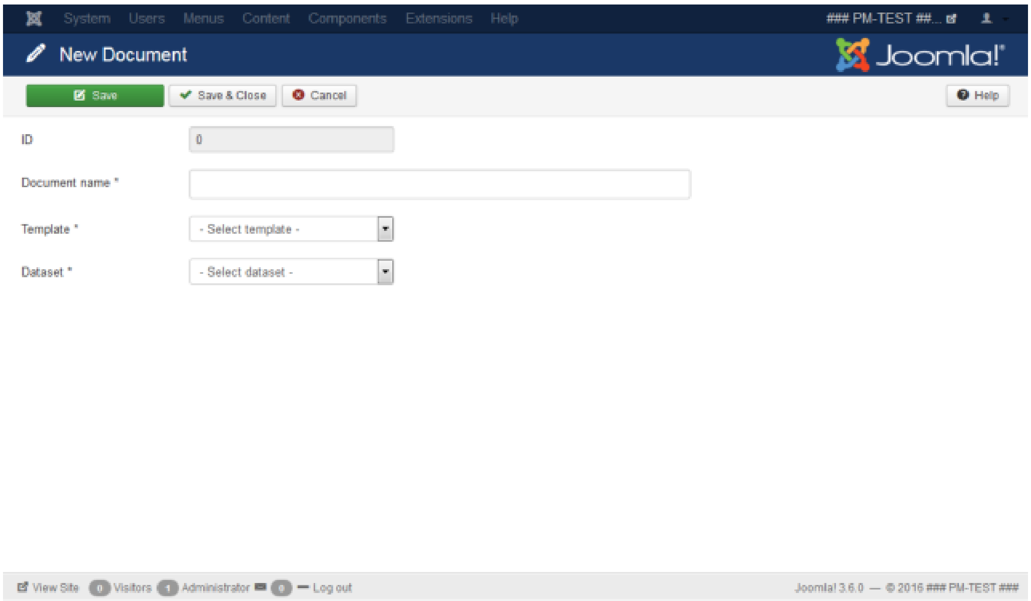
Copyright © 2016 Factory.DocWriter.Ru

- New creates a new document definition. Before creating a new document, be sure to create a dataset and a template you want to assign to this document. See Section 2, “Creating or Editing a Document”, Chapter 3, *Datasets*, Section 1, “Managing Templates List”.

- Edit. To change an existing document definition, select it in the documents list and click the Edit button. Alternatively, click the document name to open it for editing. See Section 2, “Creating or Editing a Document”.

- Delete. To delete a document definition select it in the documents list and click the Delete button.

# 2. Creating or Editing a Document



ID: 0

Document name *:

Template *: - Select template -

Dataset *: - Select dataset -

This form is used to define a document, binding together an existing dataset and template which will be used to build pages based on the database content.

- Document name. Enter unique name for a document. This name will be displayed in lists.
- Template. Select the template you want to use to build a document layout.
- Data Set. Select the dataset you want to use to get data from database.

The available command buttons:

- Save. Click to apply any changes to document parameters.
- Save & Close. Save changes to document and return back to the documents list.
- Close. Quit the document editing mode and return to the templates list. Any changes to the document parameters will be lost.

Enter document unique name, select the dataset and the template to assign to document, and click Save.

To cancel document editing and discard any changes, click Close.

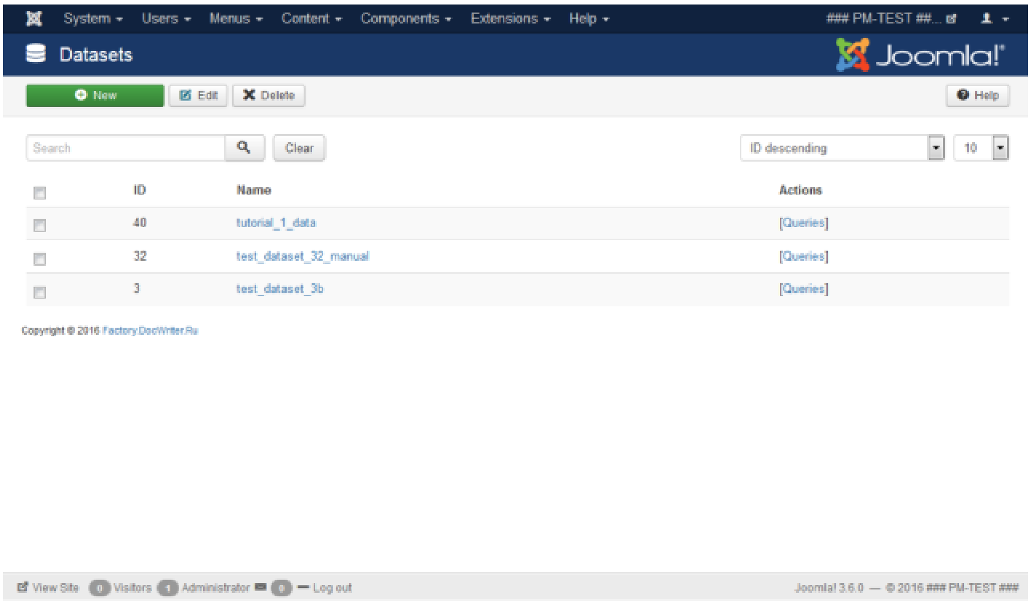
# Chapter 3. Datasets

*Datasets* are DW Anything Studio objects used to retrieve data that you want to display from database.

Each dataset can contain a number of database queries, which could be either parallel or nested. When rendering a document, DW Anything Studio first executes dataset queries, stores the results as a set of variables and passes them to a template.

## 1. Managing Datasets List

The datasets list displays all the DW Anything Studio dataset definitions.



| □ | ID | Name | Actions |
|---|---|---|---|
| □ | 40 | tutorial_1_data | [Queries] |
| □ | 32 | test_dataset_32_manual | [Queries] |
| □ | 3 | test_dataset_3b | [Queries] |

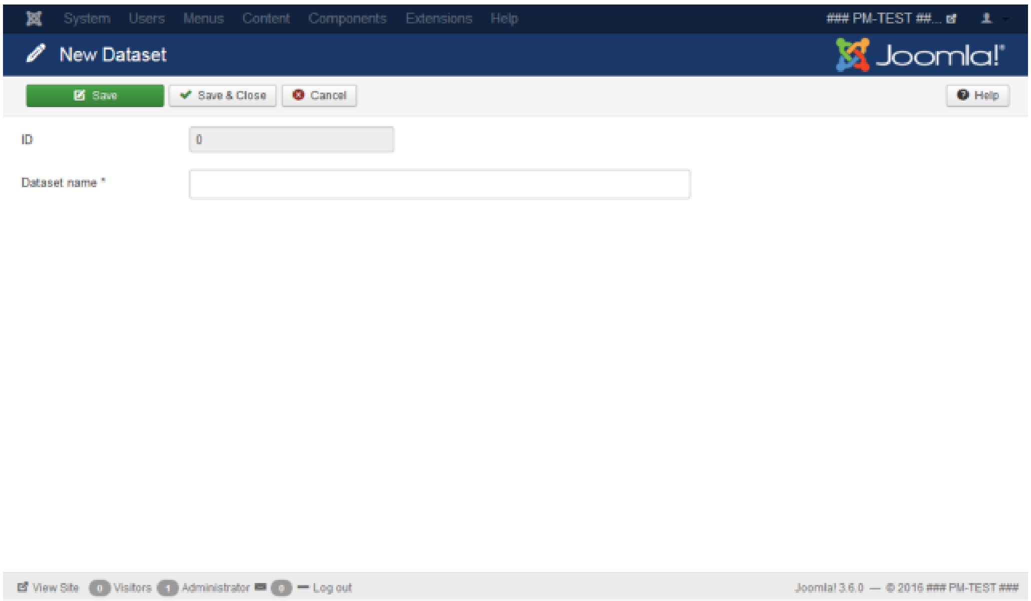
- New creates a new dataset definition.
- Edit. To change an existing dataset definition, select it in the datasets list and click the Edit button. Alternatively, click the dataset name to open it for editing. See Section 2, “Creating or Editing Dataset”.
- Delete. To delete a dataset definition select it in the datasets list and click the Delete button.

Each item in datasets list has the Actions group assigned to it. It contains buttons for all the operations applicable to the corresponding dataset.

The following actions are available for datasets.

- Queries. Displays the Dataset Queries list to manage the queries assigned to a dataset. See Section 3, “Assigning Queries to Dataset” for detail.

## 2. Creating or Editing Dataset



ID 0

Dataset name *

This form is used to define the dataset which will be used to retrieve data from database and pack it as a variables usable by a template.

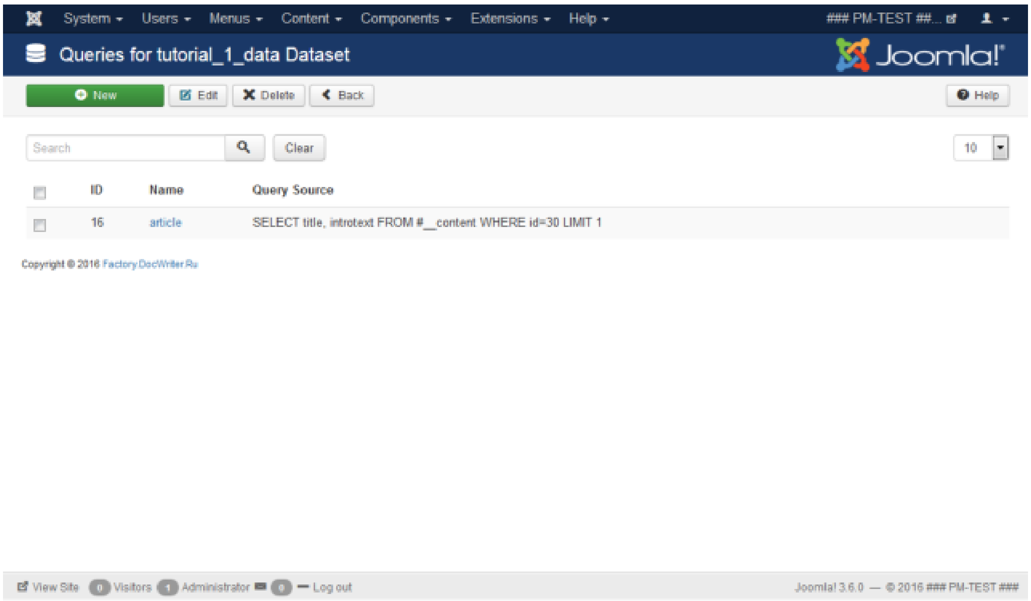
- Dataset name. Enter unique name for a dataset. This name will be displayed in lists.

The available command buttons:

- Save. Click to apply any changes to dataset parameters.
- Save & Close. Save changes to dataset and return back to the datasets list. Upon saving a new dataset for a first time, DW Anything Studio will take you to the Add Query form so you could add a query to a dataset immediately after you created it. See Section 4, “Creating or Editing Dataset Queries” for detail.
- Close. Quit the dataset editing mode and return to the datasets list. Any changes to the dataset parameters will be lost.

## 3. Assigning Queries to Dataset

The dataset queries list displays all the queries assigned to the selected dataset (the selected dataset name is included in the page heading).



Queries for tutorial_1_data Dataset

| ID | Name | Query Source |
|---|---|---|
| 16 | article | SELECT title, introtext FROM #__content WHERE id=30 LIMIT 1 |

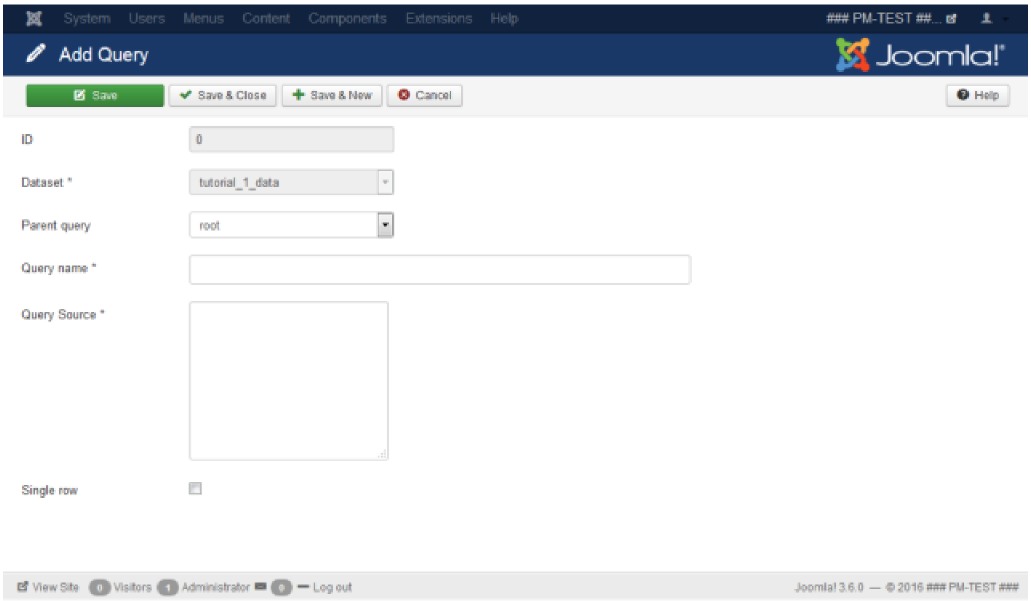
Copyright © 2016 FactoryDocWriter.Ru

0 Visitors 1 Administrator 0

Joomla! 3.6.0 — © 2016 ### PM-TEST ###

- New creates a new dataset query.
- Edit. To change an existing dataset query, select it in the queries list and click the Edit button. Alternatively, click the query name to open it for editing. See Section 4, “Creating or Editing Dataset Queries”.
- Delete. To delete a dataset query select it in the queries list and click the Delete button.
- Back. Click to go back to the Datasets list.

# 4. Creating or Editing Dataset Queries



Add Query

| | |
|---|---|
| ID | 0 |
| Dataset * | tutorial_1_data |
| Parent query | root |
| Query name * | |
| Query Source * | |
| Single row | □ |

0 Visitors 1 Administrator 0

Joomla! 3.6.0 — © 2016 ### PM-TEST ###

This form is used to define the query which will be used to retrieve a single element of data from database.

- Dataset. The pre-selected dataset to assign the query to. This is a read-only field and cannot be changed manually.
- Parent query. DW Anything Studio allows nesting queries. In this drop-down listbox, select the parent-level query for the query you are currently adding or editing.

The child query is executed for any parent query results row of data. The child query results are assigned as an array to the parent query variable named after a child query.

To avoid ambiguities, you cannot assign a query as a child to itself or any of its child queries, so these queries are not listed here.

- Query name. Query name. Alphanumeric characters only allowed. This also defines the name of the variable holding the query results. See Section 3, “Using Variables in Templates” for detail.
- Query source. Enter the SQL query to select data from database.

  For better portability, you can use Joomla! database table prefix placeholders (#__) instead of the actual prefixes.
- Single row. If checked, the query results will be stored as a single row of data. If unchecked, the query results are stored as an array of rows.

The available command buttons:

- Save. Click to apply any changes to query parameters.
- Save & Close. Save changes to query and return back to the queries list.
- Close. Quit the query editing mode and return to the queries list. Any changes to the query parameters will be lost.

# Chapter 4. Templates

*Template* is a markup of a document used to define its look and structure.

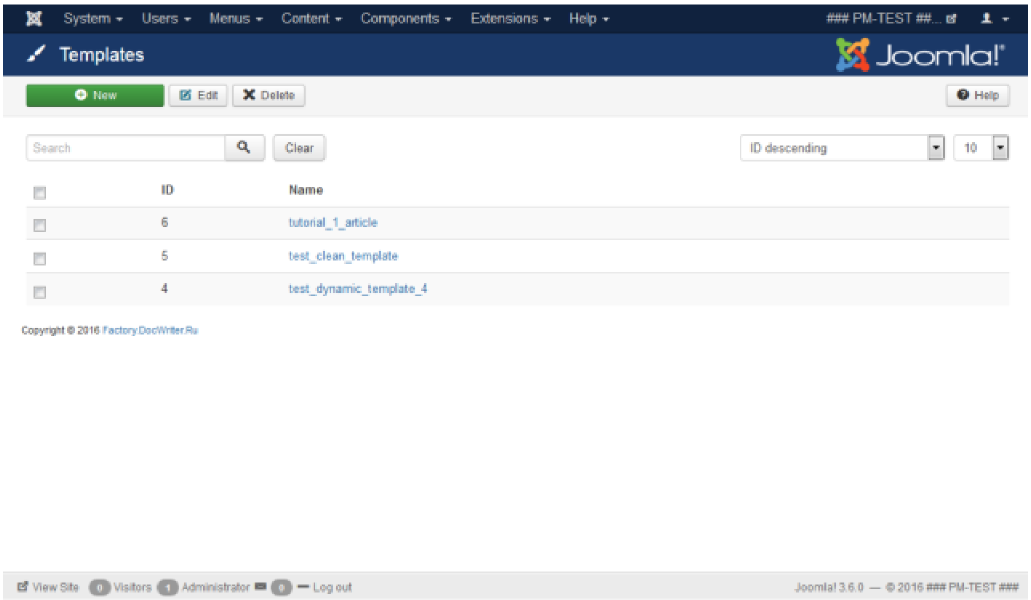
DW Anything Studio templates use the Smarty templating engine to render documents based on a data received from a dataset.

See also: Appendix C, *Smarty Implementation*.

See the original Smarty documentation for detail on Smarty templating language.

## 1. Managing Templates List

The templates list displays all the DW Anything Studio template definitions.



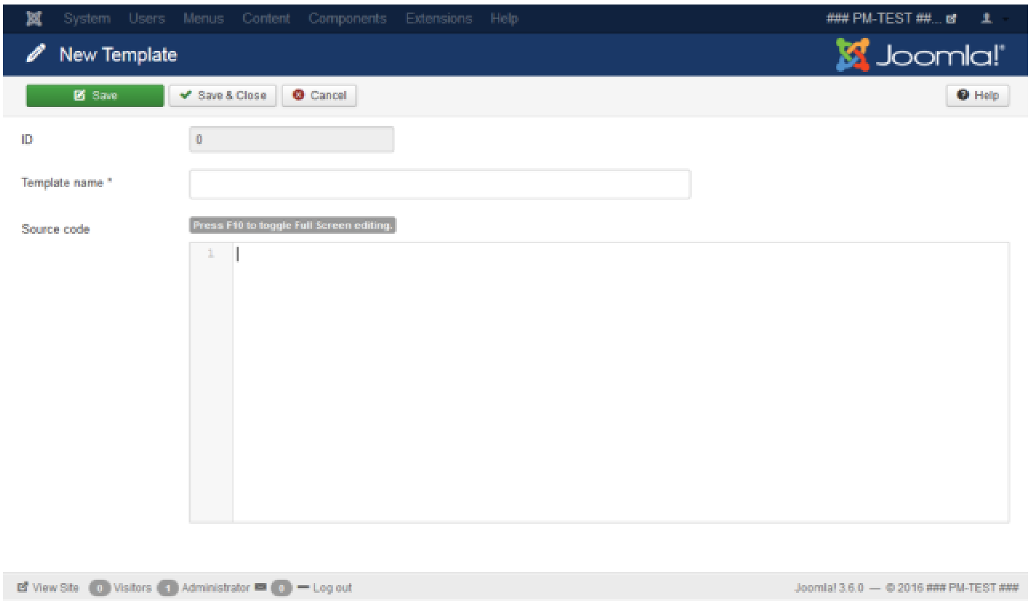
| □ | ID | Name |
|---|---|---|
| □ | 6 | tutorial_1_article |
| □ | 5 | test_clean_template |
| □ | 4 | test_dynamic_template_4 |

- New creates a new template definition.

- Edit. To change an existing template definition, select it in the templates list and click the Edit button. Alternatively, click the template name to open it for editing. See Section 2, “Creating or Editing a Template”.

- Delete. To delete a template definition select it in the templates list and click the Delete button.

## 2. Creating or Editing a Template



ID 0

Template name *

Source code

Press F10 to toggle Full Screen editing.

This form is used to define the template source which will be used to generate the resulting document.

- Template name. Enter unique name for a template.
- Source code. The template source. Contains the layout of your resulting document along with Smarty templating engine markup.

The available command buttons:

- Save. Click to apply any changes to template parameters.
- Save & Close. Save changes to template and return back to the templates list.
- Close. Quit the template editing mode and return to the templates list. Any changes to the template parameters will be lost.

See also: Appendix C, *Smarty Implementation*.

See the original Smarty documentation for detail on Smarty templating language.

## 3. Using Variables in Templates

Templates receive data from datasets as a nested set of variables.

All the available data is accessible from a template through a single top-level variable called `Data`.

To include variable value in a template text, you should use the following basic syntax:

```
{$data.section_name.variable_name}
```

See also: Appendix C, *Smarty Implementation*.

Data sections represent different types of data. See the Appendix A, *Variables Reference* for detail on data sections.

See the Appendix B, *Tutorials* for template definition examples.

See the original Smarty documentation for detail on Smarty templating language.

# Appendix A. Variables Reference

The table below describes the data sections available in DW Anything Studio.

### Table A.1. Variable data sections

| Section | Description | Usage in Template | Usage in Dataset Query |
|---|---|---|---|
| General data | The data received from the dataset query execution.<br>While all of the data hierarchy levels are available from within templates, in dataset queries you can only refer to the parent row of data | Format: `{$data.`*dataset_query_name*`.`*var_name*`}`<br>Example: `{$data.item.id}`<br>where *item* is the name of the top-level dataset query, and *id* is the column name | Format: `{`*var_name*`}`<br>Example: `{id}`<br>This returns an *id* column value from the parent data row |
| **param** | The parameters section.<br>This holds the parameters specified in the Joomla! menu item settings for DW Anything Studio document | Format: `{$data.param.`*var_name*`}`<br>Example: `{$data.param.a}`<br>where a is the name of the variable defined in the menu item settings | Format: `{param.var_name}`<br>Example: `{param.a}`<br>where a is the name of the variable defined in the menu item settings |
| **param.request** | HTTP request data.<br>Makes it possible to pass the parameters in URL and process them in your dataset query and/or template | Format: `{$data.param.request.`*var_name*`}`<br>Example: `{$data.param.request.page_id}`<br>where *page_id* is the name of the HTTP request variable (e.g.: `http://example.com/index.php?option=com_dwanything&page_id=32`) | Format: `{param.request.`*var_name*`}`<br>Example: `{param.request.page_id}`<br>where *page_id* is the name of the HTTP request variable (e.g.: `http://example.com/index.php?option=com_dwanything&page_id=32`) |

# Appendix B. Tutorials

## 1. My First DW Anything Studio Document

The following section describes how to develop the document displaying the standard Joomla! article text along with its title.

### Prerequisites

Make sure you have at least one article created in your Joomla! Content Manager.

### Defining a Dataset

First of all we should specify how to retrieve data from database. DW Anything Studio dataset will do the job.

1. Go to the Datasets list and click New to create a dataset. See Section 1, “Managing Datasets List” for detail.

2. Enter the following dataset parameters.

   **Table B.1. Dataset parameters**

   | Parameter | Value | Description |
   |---|---|---|
   | Dataset name | `tutorial_1_data` | |

3. Click Save and Close.

4. DW Anything Studio takes you to the Add Dataset Query form to define an SQL query to fetch data from database.

5. Enter the following query parameters

   **Table B.2. Query parameters**

   | Parameter | Value | Description |
   |---|---|---|
   | Parent query | `root` | This is a top-level query |
   | Query name | `article` | The article data will be stored as the `article` variable. Remember the query name must be alphanumeric |
   | Query source | `SELECT title, introtext FROM #__content LIMIT 1` | We want to get title and intro text for a single article from database |
   | Single row | `yes` | The single row of data will be stored as a variable |

6. Click Save & Close.

### Defining a Template

1. Go to the Templates list and click New to create a template. See Section 1, “Managing Templates List” for detail.

2. Enter the following template parameters.

   **Table B.3. Template parameters**

   | Parameter | Value | Description |
   |---|---|---|
   | Template name | `tutorial_1_article` | |

| Parameter | Value | Description |
|---|---|---|
| Source code | `<h1>{$data.article.title}</h1>`<br>`{$data.article.introtext}` | Our template must contain the HTML markup to display an article title and Smarty variables where we want our data to appear |

3. Click Save & Close.

## Defining a Document

1. Go to the Documents list and click New to create a document. See Section 1, “Managing Documents List” for detail.

2. Enter the following document parameters.

**Table B.4. Document parameters**

| Parameter | Value | Description |
|---|---|---|
| Document name | `tutorial_1` | |
| Template | `tutorial_1_article` | Assigning a previously defined template to a document |
| Dataset | `tutorial_1_data` | Assigning a previously defined dataset to a document |

3. Click Save & Close.

## Setting up the Menu Item

In order to access our document form front end, we need a menu item. Let's create it, following the standard Joomla! procedure.

1. Click the Main Menu — Add New Menu Item in the Menus menu of your site's administrator panel. (Note that, depending on your site configuration, there can be no Main Menu item in your Menus menu. If so, select the appropriate menu available from front end.)

2. Type `Tutorial 1 Document` in the Menu Title text box.

3. Under the Menu Item Type, click Select.

4. Select DWAnything — DWAnything Document.

5. Select `tutorial_1` in the Document drop-down list box.

6. Click Save & Close.

Now you can view your resulting document. Go to front end and click the Tutorial 1 Document menu item.

# 2. Deploying an Updates Server for your Extension

As of version 1.6, Joomla! has featured an update system. If you are an extension developer, starting from January 10, 2017 you will be required to use the Joomla! Update System in order for your extensions to be listed on Joomla! Extensions Directory.

As a part of the Update System, you must launch an updates server on your site. For detail, see Joomla! Update System Requirement in Joomla! Extensions Directory knowledge base.

This tutorial describes how to get an updates server up and running in less than 10 minutes using DW Anything Studio. The server will be connected to your downloads system to publish the updates automatically.

## The Basics

You'll need your server to render an XML document by the request from Joomla! Update System. Based upon data from your downloads system, this document should provide an information about the latest version of your extension available for download from your site.

For the purposes of this tutorial, we'll connect the updates server to the Phoca Download component, but the principles described below could be applied to the other download systems as well.

## Requirements

In order to deploy an updates server, you'll need the following:

- DW Anything Studio installed on your publicly available site
- Database client, e.g. PhpMyAdmin.

## Creating a DW Anything Studio Template

1. On your downloads site with DW Anything Studio, go to administrator panel.
2. Click Components — DW Anything — Templates to open DW Anything Studio templates management page.
3. Click New to create a new template.
4. Copy the following text and paste it into the Source code text box.

```
<updates>
    <update>
       <name>{$data.item.name}</name>
       <description>{$data.item.description}</description>
       <element>{$data.item.element}</element>
       <type>{$data.item.type}</type>
       <version>{$data.item.version}</version>
       <downloads>
           <downloadurl type="full" format="zip">{$data.item.downloadurl}</downloadurl>
       </downloads>
       <tags>
           <tag>stable</tag>
       </tags>
       <targetplatform name="joomla" version="3.6"/>
    </update>
</updates>
```

5. In the Template name box, type `update_server_xml`.
6. Click Save & close.

## Preparing Database Schema

At this point you need to analyse your downloads component database tables to check if it holds all of the information you need to fill in the server XML response as defined the template on a previous step.

You'll have to figure out, which data is already available from you downloads component and could be mapped to your template's variables directly, and which one needs to be supplied from elsewhere. To store the missing data, you should create and fill in an auxiliary table.

The table below lists all of the template variables and the results of the analysis of the `#__phocadownload`[1] table, containing the list of files available for download via Phoca Download component.

[1] Use actual database prefix instead of the `#__` characters here and below.

### Table B.5. The analysis of the `#__phocadownload` database table

| Template Variable Name (brackets and $ sign omitted) | Description | Possible Mapping from `#__phocadownload` Table | Columns Required in Auxiliary Table |
|---|---|---|---|
| `data.item.name` | The extension name as appears in the name column of the Extension Manager's Update view | `title` | |
| `data.item.description` | A brief description of your extension | Can also be a `title`, optionally combined with some general descriptive text, such as `Update of` .... This could be done via query later on | |
| `data.item.element` | The installed name of the extension. For plugins, this needs to be same as plugin attribute value for main file in plugin manifest. For `<filename plugin="pluginname">pluginname.php</filename>`, element value should be **pluginname** | | `element` |
| `data.item.type` | Type of the extension (component, module, plugin, etc.) | | `type` |
| `data.item.downloadurl` | The URL to download the extension update from | Check the format of the actual download URLs on your site and build it from data in downloads component table. | |
| | | | You'll also need an `id_phocadownload` column to link an auxiliary table record to the downloads item |

You can also include some other optional fields in your server's response XML, as described in Deploying an Update Server on Joomla! Documentation site.

Next, add the auxiliary database table `#__dwa_update_server` to your database.

1. Open your database in database client such as PhpMyAdmin etc.

2. Execute the following database query (be sure to replace `#__` with actual database prefixes used on your site):

```
CREATE TABLE `#__dwa_update_server` (
`id` INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
`element` VARCHAR(255) NOT NULL,
`type` VARCHAR(255) NOT NULL,
`id_phocadownload` INT(11) UNSIGNED NOT NULL,
PRIMARY KEY (`id`),
INDEX `FK_#__dwa_update_server_#__phocadownload` (`id_phocadownload`),
CONSTRAINT `FK_#__dwa_update_server_#__phocadownload` FOREIGN KEY (`id_phocadownload`)
  REFERENCES `#__phocadownload` (`id`) ON DELETE CASCADE
)
COMMENT='Update server definitions for DW Anything'
COLLATE='utf8_general_ci'
;
```

3. Add a record to database table to store any additional data and link it to an appropriate record in `#__phocadownload` table. Take a note of the value of an id column (primary key) of this record. It will be used to build an update server URL later on.

## Defining a DW Anything Studio Dataset

1. Go to the Datasets list (Components — DW Anything — Datasets administrator panel menu) and click New to create a dataset. See Section 1, "Managing Datasets List" for detail.

2. Enter the following dataset parameters.

### Table B.6. Updates server dataset parameters

| Parameter | Value | Description |
|---|---|---|
| Dataset name | `update_server_data` | |

3. Click Save and Close.

4. DW Anything Studio takes you to the Add Dataset Query form to define an SQL query to fetch data from database.

5. Enter the following query parameters:

### Table B.7. Updates server query parameters

| Parameter | Value | Description |
|---|---|---|
| Parent query | `root` | This is a top-level query |
| Query name | `item` | The article data will be stored as the `item` variable. Note that the query name must be alphanumeric |
| Query source | `SELECT s.*, d.title AS name, CONCAT(d.title, ' download available') AS description, d.version AS version, CONCAT('http://yoursite.com/category/', d.catid, '.html?download=', d.id) AS downloadurl FROM #__dwa_update_server s LEFT JOIN #__phocadownload d ON s.id_phocadownload = d.id WHERE s.id = '{param.request.key}'` | Note how the `downloadurl` value is formed. Form the URL according to your downloads component's URL routing rules.<br>Be sure to replace `yoursite.com` with an actual name of your downloads server |
| Single row | `yes` | The single row of data will be stored as a variable |

6. Click Save & Close.

## Defining a DW Anything Studio Document

1. Go to the Documents list (Components — DW Anything — Documents administrator panel menu) and click New to create a document. See Section 1, "Managing Documents List" for detail.

2. Enter the following document parameters.

### Table B.8. Updates server document parameters

| Parameter | Value | Description |
|---|---|---|
| Document name | `update_server` | |
| Template | `update_server_xml` | Assigning a previously defined template to a document |
| Dataset | `update_server_data` | Assigning a previously defined dataset to a document |

3. Click Save & Close.

4. Take a note of the value shown in an ID column of the documents list. This value will be used to build an update server URL later on.

## Adding Updates Support to Your Extension Package

In your extensions manifest file, within the root `extension` element, add the following code.

```
 <extension>
 <...>
 <updateservers>
    <server type="extension" priority="1" name="My Extension's Updates">
http://example.com/index.php?option=com_dwanything&amp;
id=dwanything_document_id&amp;format=raw&amp;key=id_from_aux_table
</server>
 </updateservers>
 </extension>
```

**Important!** The `server` tag must be a single line of text, no line breaks! The line breaks in this documentation are added for clarity only.

Replace `dwanything_document_id` with an id of a record with additional information stored as described in Defining a DW Anything Studio Document above.

Replace `id_from_aux_table` with an id of a record with additional information stored as described in Preparing Database Schema above.

# Appendix C. Smarty Implementation

DW Anything Studio templates inherit most of the functionality of the original Smarty templating engine, still there are some minor specific features.

These features are mentioned in the following sections.

For details on features not listed here please refer to the original Smarty documentation.

## 1. Smarty Version

This package includes Smarty version 3.10.

## 2. Limited Features

### {fetch} Function

HTTP authorisation with **user** and **pass** parameters is disabled.

This is a temporary limitation and it can be removed in future DW Anything Studio versions.

### {eval} Function and Eval Resources

Base64-encoded templates are not supported.

This is a temporary limitation and it can be removed in future DW Anything Studio versions.

### String Resources

Base64-encoded templates are not supported.

This is a temporary limitation and it can be removed in future DW Anything Studio versions.

### Request Variables

Smarty request variables based on PHP superglobal variables, such as $_GET, $_POST, $_COOKIE, $_SERVER, $_ENV and $_SESSION, are disabled in DW Anything Studio. Use the DW Anything Studio **{$data.param.request}** variable instead. See Appendix A, *Variables Reference* for detail.

This is a temporary limitation and it can be removed in future DW Anything Studio versions.