The Three File Template

If you've ever unzipped a Joomla! template and had a look at the various image folders, CSS files, javascript files, font folders, languages folders, HTML overrides (not to mention the favicon, ini and template preview files) you may be surprised to know that most of these are unnecessary.

In fact, with just a few lines of code you can turn your basic **HTML** into a **Joomla!** templated website.

In this short guide we'll create an extremely minimalist template, and we're going to use **only three files**.

The aim is to highlight the very basic essential elements required for a working **Joomla! template**.

The Three File Template

Download the source files from...
http://www.hyde-design.co.uk/3filetemplate

The XML

The first of our three files is an XML file called templateDetails.xml.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE install PUBLIC "-//Joomla! 1.6//DTD template 1.0//EN"
"http://www.joomla.org/xml/dtd/1.6/template-install.dtd">

<extension version="3.5" type="template" client="site">
  <name>3FileTemplate</name>
  <creationDate>01/05/16</creationDate>
  <author>Hyde-Design</author>
  <authorEmail>sales@hyde-design.co.uk</authorEmail>
  <authorUrl>http://www.hyde-design.co.uk</authorUrl>
  <copyright>Hyde-Design 2016</copyright>
  <license>GNU/GPL</license>
  <version>1</version>
  <description>Three file template for Joomla</description>

  <files>
    <filename>templateDetails.xml</filename>
    <filename>index.php</filename>
    <folder>css</folder>
  </files>

  <positions>
    <position>logo</position>
    <position>footer</position>
  </positions>

</extension>
```

This maps out specific details of your template, including the template **name**, your **author** name, **licence**, **copyright** notice, **year** and a short **description** of your template. None of these details appear at the front end on the live site, they are only visible in the template manager of the Joomla! admin panel.

The XML lists all the files belonging to the template so that they can be installed correctly. In this example we only have three files:

- the css folder (where we keep the **editor.css** file)
- the **templatedetails.xml** file itself
- the **index.php** file.

Each filename is wrapped in a **<filename>** tag and each folder is wrapped in a **<folder>** tag. By including a folder name there is no need to individually include each filename within that folder.

The XML file also declares which **module positions** you want your template to include (these will appear in the drop down box on the module manager but don't worry if you forget to add any positions, they can just be typed in).

The PHP

The index.php file looks like a normal HTML file with a <head> and <body> section.

```
<!DOCTYPE html><html lang="<?php echo $this->language; ?>">

<head>
<meta charset="utf-8" />
<jdoc:include type="head" />
<link rel="stylesheet" href="<?php echo $this->baseurl ?>/templates/
<?php echo $this->template ?>/css/editor.css" type="text/css" />
</head>

<body>
 <div id="container">
  <div id="logo"><jdoc:include type="modules" name="logo" /></div>
  <div id="content"><jdoc:include type="component" /></div>
  <div id="footer"><jdoc:include type="modules" name="footer" /></div>
 </div>
</body>
</html>
```

We're using HTML5 markup so the doctype is nice and small.

The **<html>** tag is opened after the doctype and closed with a **</html>** tag at the end of the file.

A language attribute is included within the opening **<html>** tag to inform browsers and search engines which language is used. The **language** value itself is inserted using PHP to pull the settings from the Joomla! configuration.

In the **<head>** section you'll notice there are no page titles, meta-keywords, meta-descriptions, meta-generator tags, etc that you would normally find in an HTML file. Remember this is a template that will form the basis of MANY DIFFERENT PAGES, if we added in a title here it would appear the same on every page (and we don't want that). So the line...

<jdoc:include type="head" />

...is a Joomla! statement that inserts all the meta tag info from the current Joomla! article or menu page settings.

Also in the **<head>** section we have a reference to our external CSS file.

The path to this file includes two snippets of PHP, one of them states the **base URL** and the other one states the **template name**. This means you don't have to change this line each time you install the template on a different domain or each time you decide to rename the template.

index.php

The main part of the template is in the **<body>** section. You'll see that there are three divs (**logo**, **content** and **footer**) wrapped in another div called **Container**.
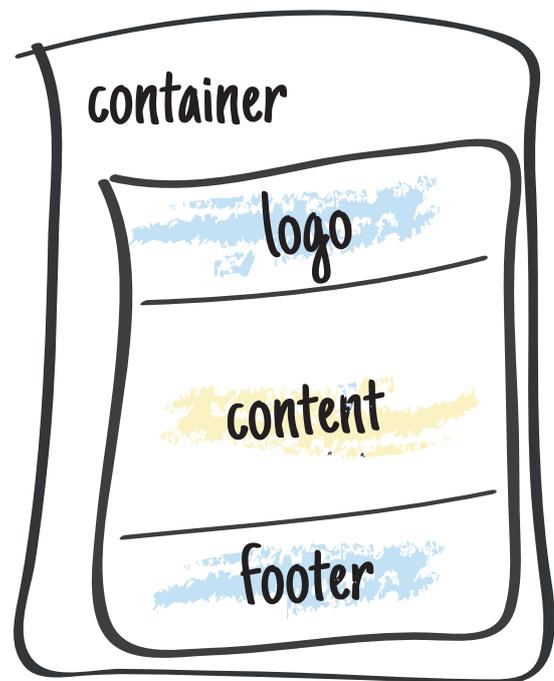
The **content** div is where the main joomla article or **component** content is displayed. Between these div tags is the statement...

<jdoc:include type="component" />

The **logo** and **footer** divs are both going to display module content. So between those div tags is the statement...

<jdoc:include type="modules" />

You'll see that each module declaration includes a name (such as **name="logo"** or **name="footer"**), this corresponds to the module position, so if a module is set to appear in the footer position, it will only render to that named statement.

container

logo

content

footer

You can create as many extra module positions as you want and if you're used to designing layouts in Dreamweaver then you can copy the HTML code in and simply insert the **jdoc statements** at the relevant points...

...just don't forget to also include the component statement otherwise there'll be nowhere to display your articles.

5

The CSS

There's nothing particularly Joomla!-centric about the editor.css file.

```
@charset "utf-8";
#container {width:980px; margin: auto;}
#logo, #content, #footer {width: 100%; height: auto; text-align: left; }
```

As you can see the CSS rules are fairly minimal, but when you build your site this file in particular will grow as you are likely to want to style headings, hyperlinks, module positions, backgrounds and set different classes for different text content.

This first line of the CSS file is a standard instruction to treat the file as UTF-8 encoded (so that browsers will render the code in the correct format).

The second line sets the **container** div width at **980px** and gives it an equal margin to the left and right.

The last line sets the width and height of our **logo**, **content** and **footer** divs and aligns the text to the left. They currently all share the same height and width values but this is something you'll probably want to change.

← 980px →

The set width of 980 pixels ensures the site can be viewed on smaller monitor screens without the annoyance of scrolling from left to right.
The fact that the main div is centred means that on larger monitors there will be plenty of white space to either side (which can be filled with a background image, colour or pattern).
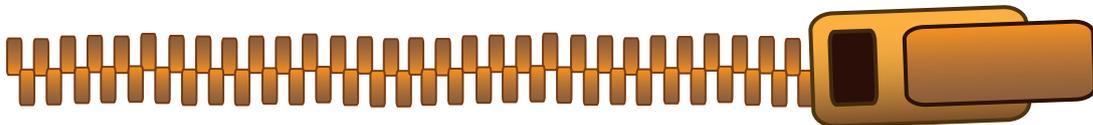
There's nothing particularly Joomla! centric about this CSS file apart from it's filename...

Joomla's default **WYSIWYG** editor automatically searches for a CSS file called **editor.css** and includes any styles within the editor page and any classes within the toolbar. So the styles declared here will actually be used when editing articles.
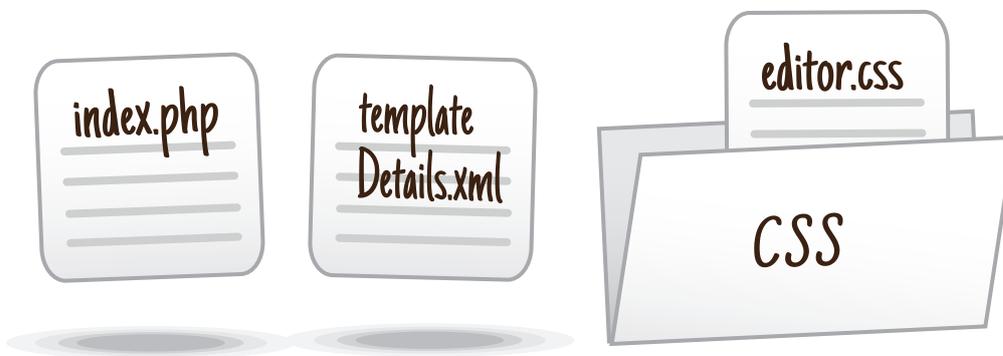
If you want to avoid this you can give the CSS file an alternative title or use more than one file (useful for seperating the typography from the layout elements.)

Zipping it up

To turn your three files into a template they simply have to be zipped up.

Joomla! requires the css file to be placed in its own folder (called "css"), so before you zip the files up check that the file / folder structure is the same as below...

As soon as this is zipped up it can be installed via Joomla's **Extension Manager** (as per any other template).

Don't forget you can still edit the i**ndex.php** file and the **editor.css** file after it has installed via the Joomla! **Template Manager**.

All the missing bits

There are many additional files in a Joomla! template that are deliberately not included in the "3 file template" as they are not strictly essential. They are of course very useful, so here's a rundown of what they do...

index.html

This is a blank file that is included so that people do not try to access the template folder via a web-browser. Adding the **index.html** file keeps the site secure by displaying a blank page instead of a list of files and folders. If you create any new folders within your template always include a blank **index.html** file too.

favicon.ico

The **favicon** is the small "shortcut" or "bookmark" icon often displayed at the top of your browser window. It is usually 16 x 16 pixels and in order for it to be included it should always be titled **favicon.ico**

error.php

If someone enters a misspelt URL or clicks on an outdated bookmark link then the **error.php** file is loaded instead of the **index.php** file. The standard **error.php** file is stored in the **templates > system** folder and can be edited and copied into your main template folder, allowing you to customize those 404 errors.

template_preview.png   template_thumbnail.png

The **template_thumbnail** image is the graphic that represents your template (as displayed on the template manager). The **template_preview** image is the enlargened version that appears when you click on the thumbnail image.

params.ini

Template parameters are values that are set up in the **templateDetails.xml** file and can be edited via the Template Manager to do all sorts of things (eg. changing a background colour, switching from fluid to fixed widths, altering font sizes, etc). The current parameter values are stored in this small params.**ini** file instead of in the database.

This template doesn't use any background images but you probably want to. The **images** folder is therfore a tidy storage place for all the images referenced in your css file. Don't forget you'll need to apply the **../images/** prefix to the file-name so that the file can be correctly located from this folder. For example...

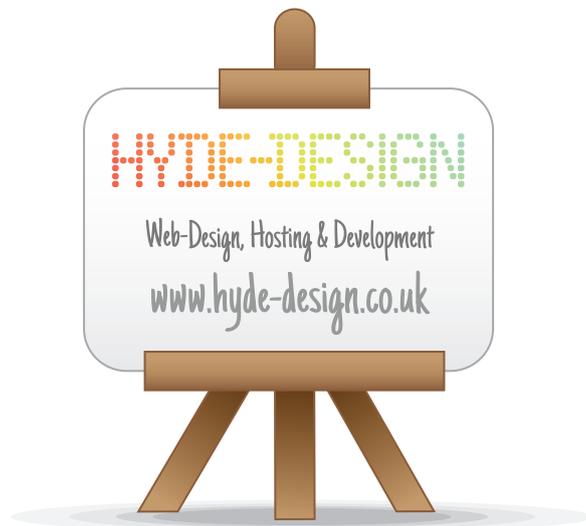#logo {background-image: url('../images/logo.jpg');}

As with the images folder, the **javascript** (or "js") folder simply stores any javascript files that may be used in your template.

The **languages** folder. contains translations of any static text used by the template. Each language contains a sub-folder (eg. the English language is titled "en-GB") and there are seperated files for front-end translation and admin translation.

The **html** folder is one of the most powerful features of the whole Joomla! framework. If you need to change any of the core Joomla! files (for example: to add code to the category blog page or to change the layout of a standard module) adding these files into the templates html folder ensures they are loaded instead.

HYDE-DESIGN

Web-Design, Hosting & Development

www.hyde-design.co.uk

**If you found this guide useful, please feel free to visit us at
wwww.hyde-design.co.uk**