

## razorCMS Theme Development Guide

### Creating a theme

To create a new theme for razorCMS, there are a few things that need to be explained first, mainly the interface for introducing the theme into the system, as well as the handles and function calls that exist in the theme itself that integrate the system into the xhtml template or public facing front end as I like to call it.

### Installing new themes

Themes are added to razorCMS in just the same way that any add-on is added to razorCMS. By using the add-on system called the blade pack interface, all themes can be uploaded into the blade\_packs directory and activated using the blade manager in the simplest of ways. This makes adding new themes a breeze and gives you the ability to switch them off and on with a single click.

### The theme blade pack structure

It is very important that the structure for creating a new theme is correct, this ensures good compatibility with the system and more importantly with other add-ons.

A theme pack is split up into several parts, the blade pack connecting the theme to the system, an XHTML template file used for the html structure for the site, the CSS template containing all style information for the website front end, an editor css file to match the front end theme and any images or script files that are needed.

The file structure of a theme blade pack is as follows

```
public_theme_themename.php
public_theme_themename
|----- themename_xhtml.php
|----- themename_css.css
|----- editor_css.css
|----- any other files
```

public\_theme\_themename.php

This is the blade pack used to integrate the theme into the system.

public\_theme\_themename

This is the folder where all theme blade pack files are placed.

themename\_xhtml.php

This is the html/xhtml template file, this can be written in xhtml or html, in either strict, transitional or frameset, adjust the filename accordingly to suit. Ensure any work is standards compliant.

themename\_css.css

This is the style file, it contains all style information regarding the theme, this is stored as css in a filename ending in '.css'. Ensure any work carried out is

standards compliant.

#### editor\_css.css

This is the style file for any WYSIWYG content editors, it contains a cut down version of the themename\_css file, enabling the editor to show content in the editor, as though it was displaying just the content part of the main theme, this is stored as css in a filename ending in '.css'.

#### Any other files

This is any image files used in your theme, and javascript files that contain any special scripts.

### Creating a new structure

The easiest way to get started is to download the sample theme blade pack archive, this is a template set on which all new blade packs can be built. The archive contains a sample blade pack connector and a sample folder containing a sample XHTML and both CSS template files.

Once you have downloaded the sample archive and unpacked it, the first port of call should be to think of a name for your theme, try to make this original and unique, once you have this rename the blade pack connector public\_theme\_sample.php changing the name 'sample' for the name of your theme.

Once this is complete, do the same for the folder, and the files within the folder sample\_xhtml.php, sample\_css.css, you can leave editor\_css.css as is. When creating your theme, any images or scripts used in the theme should be stored in this directory too.

Now we nearly have our structure in place we have one last task to complete, we must alter the public\_theme\_sample.php file that you renamed just a minute ago, we need to change a few things in order to integrate the theme blade pack into the system.

Open the file that you renamed and change the following things.

First alter the notes at the top of the page updating the information so we know the details of the theme

```
////////////////////////////////////  
// Blade Pack Theme //  
// Theme Name //  
// License type //  
// Date //  
// Author //  
////////////////////////////////////
```

Ensure you keep the layout the same to keep uniformity throughout blade packs.

Next alter the blade pack details to reflect the name, your name and a brief description.

```
////////////////////////////////////  
// Blade Pack Details //  
////////////////////////////////////
```

```
// Register Blade Pack Name //
$bladePack = new BLADEPACK( 'Public Theme - Theme Name
Goes Here' );
// Register Blade Pack Author //
$bladePack->Author('Author Goes Here');
// Register Blade Pack Description //
$bladePack->Description( 'Public facing theme - Theme
description goes here' );
```

Only alter the text in between the single quotes.

Next we alter the blade names, this will tell the blade interface which blades to run in which sockets within the code, they correspond with the blades further down the code. Alter the text 'themeName' in the following bit of code, swapping it for your theme name. If you name is more than a single word, any other words start with a capital letter.

```
////////////////////////////////////
// Socket Allocation //
////////////////////////////////////

// Add Blades to Sockets addBlade( 'socket', 'blade' ) //
$bladePack->addBlade( 'public-change-theme',
'themeNameXHTML' );
$bladePack->addBlade( 'public-css-address',
'themeNameCSS');
$bladePack->addBlade( 'editor-css-path',
'themeNameEditorCSS');
```

The last thing we need to do is edit the blades themselves, there is two in total, one to integrate the xhtml template file and one to integrate the css template file.

Alter the text 'themeName' in the following code to match the changes you just made above so the blades are picked up by the sockets allocated above. Ensure you match the case correctly as these are case sensitive.

```
////////////////////////////////////
//      Blades      //
////////////////////////////////////

// blade - load new xhtml template //
function themeNameXHTML(&$xhtmlTemplate) {
    $xhtmlTemplate =
'blade_packs/public_theme_sample/sample_xhtml.php';
}
// end //////////////////////////////////////

// blade - load new css template //
function themeNameCSS(&$cssTemplate) {
    $cssAddress = explode(RAZOR_CSS_FILE, $cssTemplate);
    $cssTemplate =
```

```
$cssAddress[0]. 'blade_packs/public_theme_sample/sample_css.css';  
}  
// blade - load editor css template //  
function themeNameEditorCSS(&$cssfile) {  
    $cssfile =  
    'blade_packs/public_theme_sample/editor_css.css';  
}  
// end ///////////////////////////////////////  
  
// end ///////////////////////////////////////
```

Once you have changed this, you need to alter the text 'blade\_packs/public\_theme\_sample/sample\_xhtml.php' to match your template xhtml file. Change the two instances of the word 'sample' to match the folder and file we renamed right at the start.

Once you have altered the xhtml template, you need to do the same for the css template in the next blade. Alter 'blade\_packs/public\_theme\_sample/sample\_css.css' to match your template css file.

The last change you have to do is alter the editor theme location, leaving the editor filename as is. Just alter the folder public\_theme\_sample to match the lines above

'blade\_packs/public\_theme\_sample/editor\_css.css', change sample to match the other lines above.

Change the 3 instances of the word 'sample' to match the folder and file we renamed right at the start.  
After all the changes have been made, you can save the file and your structure is complete.

### Writing your theme

Next you need to write your theme, now unfortunately I can't help you with that, it is down to you and your artistic flair. But what can do is give you some advice on the system and how it integrates.

The three files you want to alter are the xhtml/html template file, the css template file and the editor css file, leave the editor until last, as you can do this by copying your main css file pasting it into your editor css file and simply changing it to match when using an editor. These are where your theme code should be placed. Any images or scripts used in creating the theme should also be placed in the same folder as the template files.

When creating your xhtml template, there are several function calls and sockets that need to be placed in the template, this will ensure that all information is displayed in the public front end, and that any add-ons are compatible with your theme.

Whilst it is ok to leave out any of the following function calls as these can omit

things like top navigation, something that is not always desired, it is recommended that all sockets be placed in your template. Failure to include a socket could result in an add-on not functioning with your template. Please ensure that all code is standards compliant, themes written will be verified for compliant XHTML/HTML and CSS, only those passing will be included into the main repository of blade packs.

### Function calls

The following function calls are used to display data within the template.

```
<?php loadSettings('sitename'); ?>
```

Loads the name of the website, do not alter this code when using it. Can be used in the title area of a template and anywhere you wish to display the website name.

```
<?php loadPageTitle(); ?>
```

Loads the name of the page loaded at that time, do not alter this code when using it. Can be used in the title area of a template and anywhere you wish to display the loaded page title.

```
<?php cssLocation(); ?>
```

Is used for the reference to the css file, do not alter this code when using it. This corresponds to the css file location that was altered in the blade pack connector earlier. Use this as the css file reference in the link command, place it in the href part of the link declaration.

```
<?php loadSettings('siteslogan'); ?>
```

Used to display the site slogan, do not alter this code when using it. This will display the website slogan where ever it is placed in the template.

```
<?php loadLinks('categoryName'); ?>
```

Use this to display category links from any given category, change the text 'categoryName' to the name of the category  
Your pages are stored under. Default category names that ship with the core download include 'top-navigation' for navigation across the top of the page, 'sidebar' for displaying a side navigation and 'footer' for displaying footer navigation.

```
<?php loadSlabContents(); ?>
```

Use this to display the page contents, this should only be used once on a page, do not alter the code when using it. Place this where you wish to have page content displayed on your site.

```
<?php loadInfoContents(); ?>
```

Use this to display the info column on your site, this is the place that all page content saved in the infobar category are placed when allocated to a page within the content manager. Do not alter this code when using it.

```
<?php loadSettings('copyright'); ?>
```

Use this to display the copyright data that is set in the settings manager. Do not alter this code when using it.

### **Sockets**

The following sockets are important, they must be placed into the xhtml template in the correct place to ensure all add-ons are compatible with your template.

```
<?php BsocketB('public-xhtml-head1'); ?>
```

Place this socket on the line before the css link declaration in the head of your html/xhtml document.

```
<?php BsocketB('public-xhtml-head2'); ?>
```

Place this socket on the line after the css link declaration in the head of your html/xhtml document.

```
<?php BsocketB('public-xhtml-header'); ?>
```

Place this socket in the header part of your html/xhtml document, ensure it comes after your website name or logo.

```
<?php BsocketB('public-xhtml-topnav'); ?>
```

If you have a top navigation, place this on the line after the function call to load the navigation links, if you do not have a top navigation in your template, place this as the last line in your header section.

```
<?php BsocketB('public-xhtml-content'); ?>
```

Place this socket on the line directly after the function call to load the page content.

```
<?php BsocketB('public-xhtml-leftnav'); ?>
```

Place this socket on the line directly after the function call to load any sidebar navigation. If your template has no sidebar navigation, place this socket before the above content socket.

```
<?php BsocketB('public-xhtml-leftbar'); ?>
```

Place this socket on the line directly after the function call to load info contents. This is normally placed in the side bar, but can be placed anywhere

the info contents are loaded.

```
<?php BsocketB('public-xhtml-footer'); ?>
```

Place this socket as the last line in the footer section of your template.

```
<?php BsocketB('public-xhtml-endofdoc'); ?>
```

Place this socket on the line above your closing body tag </body>.

If you are at all unsure of the placement of any sockets when writing your theme blade pack, please visit the support forum for help and advice. All sockets will be checked before adding a theme blade pack into the main repository.

### **Optional calls**

Display site name as a link to the home page

To display the main site name at the top of your page, with a link back to the home directory location, use the following call.

```
<h1><a href="<?php echo $_SESSION['PHP_SELF']; ?>"><?php  
loadSettings('sitename'); ?></a></h1>
```