razorCMS Development
Extensions
Themes
(V3.X.X)

Author	**Role**	**Handle**
Paul Smith	Author	smiffy6969, razorcms, razor

Revision	Changes	Date
1.0	Initial creation on Google Docs	03/2014

Contents

razorCMS Extension Development

Preface

This document is supposed to be an introduction to creating themes in razorCMS V3.X.X and is primarily to serve as a way to get the process out there until it can be sucked into a website, in a clear manageable way. Until such time, all extension development can be reference in the development docs.

Introduction

What is a theme extension?

A theme extension is simply a specific type of extension used to control themes in razorCMS. The themes are generally split into two main areas, system and theme. system extensions come in lots of areas, like communication. Themes are a simply a specific area of extension type that we handle separately to other extensions.

Theme extensions contain all the required files needed to run a theme, all in one nice little package. Themes may include style (less, css, please note less files must be compile manually but bundled for easily changing style later), php, views (php files with html and php page structure), images, javascript and finally a manifest (json file to detail the extension details, information on layouts and a means of identifying each theme/layout).

A standard theme should consist of at least the following, in this folder layout:-

```
extension
    theme
        handle (such as ‘razorcms’)
            theme-name (your theme name separated by hyphens)
                images
                    (any image files used by any of the layouts)
                style
                    (any less and css files used by any of the layouts)
                view
                    layout-name.php (your themes layout as detailed in manifest)
                    layout-name-2.php (you may have multiple layouts)
                layout-name.manifest.json (Your first layouts manifest file)
                layout-name-2.manifest.json (any other layout manifests)
```

This is the most basic structure showing you the ability to use multiple layouts for each theme. In fact you are encouraged to created at least 3 or more variations for people to choose from. Each layout may have it’s own style, images etc, but it is much more conservative to create a master style file, then use bootstraps grid system for setting the layout up. This approach works well and cuts down on the amount of separate styling. Also it is a good idea to remember we are mobile friendly, please ensure your system uses the bootstrap layout markup in order to play nice with mobiles.

Getting started with creating themes

Before you can publish any themes on the razorCMS website, you will have to register your handle. Until such time I can get a nice registration page set up, I suggest we do this ad-hoc via email, I can be contacted via the website contact form, so until then lets do it this way.

Drop me an email stating your handle, if it is available I will reserve it for you, this will be a means as to identifying and grouping extensions together. You don’t have to do this, of course, but if you wish to publish your theme (and get some free advertising for your site) on the razorCMS site I will only do this for those with valid handles. The only exceptions to handle names are, not to use razorcms, razor, smiffy6969, administration or admin, these are my reserved names. Lets try to keep this clean people, I will not publish rude, offensive or derogatory handles.

Once you have your handle registered with me, you are more than welcome to pass on any themes, I will then publish these to the razorCMS website as I can find the time.

The best way to start creating your theme is to take one that already exists and copy it, for this to work we can use any theme we like. What better process, than taking a theme that closely resembles what you want layout wise, then alter it. Using thismethod I am able to create themes in half an hour or less. I will demo this process here based on the default-extra theme, designed to be an addition to the default theme, but you could use any of the available themes from the razorCMS website.

Altering an already existing theme

The first thing we need to do is copy the theme to a new location. On your desktop (or wherever you like for that matter). Download the default extra theme from the website, extract it to desktop, and we can start to configure it.

1. Alter the file structure

You will notice the default-extra theme is saved in the folder named razorcms. This is the handle I use for extensions. The handle should be registered with razorCMS if you want to publish on the site, for now we will give you the default handle ‘dave’.

Change the razorcms folder name to ‘dave’.

Next we need to give the theme a name, we will call this ‘daves-new-theme’. Change the folder inside the ‘dave’ folder called ‘default-extra’ and change this to ‘daves-new-theme’.

You should now have the following structure.

```
extension
    theme
        dave
            daves-new-theme
                images
                style
                    style.css
                    style.less
                    style_ie8.css
                    style_ie8.less
                    style_ie9.css
                    style_ie9.less
                view
                    1-column.php
                    2-column.php
                    3-column.php
                1-column.manifest.json (matches the 1-column layout)
                2-column.manifest.json (matches the 2-column layout)
                3-column.manifest.json (matches the 3-column layout)
```

2. Configure the manifests

We now need to configure the manifests (this is a three layout theme), with your details. Open each manifest file and change the following details. So you will need to repeat this step three times for 1-column.manifest.json, 2-column.manifest.json and 3-column.manifest.json.

For the timestamp, ‘created’, visit http://www.unixtimestamp.com/ and get a new timestamp

Change these values, leaving the rest as they where:-

```
{
	"handle": "dave",
	"extension": "daves-new-theme",
	"name": "Daves New Theme",
	"author": "dave",
	"version": 1,
	"created": 1394394028,
	"description": "This is daves new theme and it’s really good. It has 3 layouts."
}
```

As detailed earlier, you will need to configure all three manifests (ensure you do not mess up the layout names).

This is all you will need to change in order to configure the manifest, for a details description of all fields, see below:-

manifest: This is the manifest file name, which matches the layout name, place in quotes.

type: This is the name of the extension area, for themes, it’s theme, place in quotes.

handle: Your handle, registered with razorCMS if you want to publish to our website, place in quotes.

extension: Your theme name, place in quotes.

layout: The layout name, which should match manifest name, and php view name, place in quotes.

name: A nice readable name for your theme, please write as a title, place in quotes.

author: Your handle, as you registered it, place in quotes.

version: Your version number, please use whole integers (e.g. 1,2,3…), no quotes.

created: Time stamp of creation date (as above, in red), no quotes.

description: A good description of your theme, please be infromative, 500 words max, place in quotes.

menus: ["header","footer"], An array of menus used in this layout.

content: {"header": 1,"main": 1,"footer": 1} An object detailing location columns (how many cols per location).

3. All done, now make your theme look how you want it to

Now you have your basic layout and manifests configured, you should be able to install this theme into razorCMS. You can install the theme as you would any theme, just simply copy the extension folder that contains the theme, directly into the web root of your razorCMS install, merging any folders.

Once installed, login to your website by appending /login to the end of the URL, choose a page to use the new theme on (the second page on a fresh install has some good test output on it), click the dashboard icon and select the new theme and save. Once loaded, you may now start altering the view files to configure the template, and altering the less/css files to configure the style.

Please note, we try to use less to write style sheets, it’s nicer, cleaner, easier to read and has lots of benefits. Ensure you compile any less files to css before testing, and please supply less files with your themes.

The basics of themes in razorCMS V3

Themes are generally split into three parts, you have the views (the layout PHP files), you have the style (we write less and compile to css to be used in the layout views) and we have the accompanying assets (js, images etc.).

Views

These are PHP files, generally a html template view created as a PHP file to take advantage of server side processing, like setting base paths, outputting menus, content and other fun stuff. The file is laid out much like a HTML file with the addition of PHP tags in certain places.

There are a few things you need to be aware of with the views, like the use of a custom opening body tag. This is very important and the basis behind allowing the angularJS overlay to be inserted in for managing the website. If you miss this out, you will have no admin panels.

Instead of having an opening body tag, we use the following, then close it normally as follows:-

```
<?php $this->body() ?>
</body>
```

The next is that we require certain attributes on the html tag, to help angularJS out

```
<html xmlns:ng="http://angularjs.org" id="ng-app">
</html>
```

Finally, we require certain assets loaded in order to make the system function correctly, as well as specific meta tags set. Please ensure any of your assets come AFTER the default assets.

```
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="<?php echo $this->page["description"] ?>">
    <meta name="keywords" content="<?php echo $this->page["keywords"] ?>">

    <title><?php echo $this->site["name"] ?>::<?php echo $this->page["title"] ?></title>
    <link
href='http://fonts.googleapis.com/css?family=Open+Sans:400italic,600italic,700italic,800italic,400,700
,800,600' rel='stylesheet' type='text/css'>

    <!-- resolve base URL/IP/U-AGENT for any js applications -->
    <script type="text/javascript">
        var RAZOR_BASE_URL = "<?php echo RAZOR_BASE_URL ?>";
        var RAZOR_USERS_IP = "<?php echo RAZOR_USERS_IP ?>";
        var RAZOR_USERS_UAGENT = "<?php echo RAZOR_USERS_UAGENT ?>";
        var RAZOR_PAGE_ID = "<?php echo $this->page['id'] ?>";
    </script>

    <!-- require js -->
    <script <?php $this->data_main() ?> src="<?php echo RAZOR_BASE_URL
?>library/js/require.js"></script>
    <script src="<?php echo RAZOR_BASE_URL ?>library/js/require-config.js"></script>

    <!-- load bootstrap, style overrides and public css -->
    <link type="text/css" rel="stylesheet" href="<?php echo RAZOR_BASE_URL
?>library/style/razor/razor_base.css">
    <!--[if IE 9]><link type="text/css" rel="stylesheet" href="<?php echo RAZOR_BASE_URL
?>library/style/razor/razor_base_ie9.css"><![endif]-->
    <!--[if IE 8]><link type="text/css" rel="stylesheet" href="<?php echo RAZOR_BASE_URL
?>library/style/razor/razor_base_ie8.css"><![endif]-->

    <!-- load your theme style here -->
    <link type="text/css" rel="stylesheet" href="<?php echo RAZOR_BASE_URL
?>/extension/theme/razorcms/daves-new-theme/style/style.css">
    <!--[if IE 9]><link type="text/css" rel="stylesheet" href="<?php echo RAZOR_BASE_URL
?>/extension/theme/razorcms/daves-new-theme/style/style_ie9.css"><![endif]-->
    <!--[if IE 8]><link type="text/css" rel="stylesheet" href="<?php echo RAZOR_BASE_URL
?>/extension/theme/razorcms/daves-new-theme/style/style_ie8.css"><![endif]-->
</head>
```

So as you can see there is a lot to ensure you get right. To be honest, if you copy the default-extra theme, you only need to remember to change 3 things highlighted in red above, thats the location to your css files.

You will notice a few things in the head section. Like we are using php to load certain values in meta data, using it to configure the load point for requireJS (to load the admin overlays), we use constants to get the base paths and base urls, as well as using conditional comments to add to themes for IE8 and IE9 (Yes I know we do not support IE8, but others may want to support it for the publicly viewable site, not the admin overlays). Use the IE override style files to tweak the theme for IE8 and 9. IE10 and up should be pretty compatible.

Moving on to the body area, again, don't forget that custom body opening tag, I cannot stress this enough. If you are not aware of bootstrap, I would get used to it now http://getbootstrap.com/ as it will save you the most amount of time ever, I am serious, learn it. Once you have learnt it, you may better understand the markup in the body area.

We use version 3 and up of Bootstrap, so use your new powers to set out the theme as you wish, use the power of the grid system to layout your page and make it mobile friendly to.

The only real thing I need to point out here is how to load the menus and content.

```
<ul class="nav nav-pills mobile-hide-block">
    <?php $this->menu("header"); ?>
</ul>
```

This will load a header menu, if one is present in the system. The three main menus are 'header', 'main' and 'footer'. At present the ALPHA version only allows these menus to be used, but the BETA version will automagically create any new menus you add. Basically, if it cannot find the menu, it will create it for you. This will allow you to add menus simply by adding the snippit above in your view.

For the interests of interoperability across themes, please use header, main and footer for main navigation. Only result to adding custom names for 'special' navigation. This will ensur ethat when people swap themes, they do not lose all navigation.

The next thing to point out is this.

```
<?php $this->content("main", 1); ?>
```

This little snippit will load content, it will load any content from the main area column 1. The content locations are split into areas and locations. The three main areas are header, main and footer. You may only use these locations. However you may have as many columns as you wish. Please order columns in the order of importance. By this I mean place column 1 in the most important place of the area. For the 'main' area, column 1 is usually the largest content area. Columns 2 and 3 would probably be left or right side bars.

Again you may have as many as you like

```
<div class="row">
    <div class="col-sm-6">
        <div class="template-main-content">
            <?php $this->content("main", 1); ?>
        </div>
    </div>
    <div class="col-sm-6">
        <div class="template-main-content">
            <?php $this->content("main", 2); ?>
        </div>
    </div>
</div>
```

Thats it really, you should be able to layout your view now in a bootstrap friendly way and get the most out of bootstrap, overriding the default look and feel for something more unique.

The last thing I want to finish on here is the default style. We first load the bootstrap styling, extend this (we could use mixins here, I may swap to this later at some point to be able to alter mixins, this may make it to BETA). You should be able to use the following styles for controlling what you show or hide for mobile/desktop.

```
// show on mobile
.mobile-show-block
.mobile-show-inline-block
.mobile-show-inline

// hide on mobile
.mobile-hide-block
.mobile-hide-inline-block
.mobile-hide-inline
```

These classes should allow you to manipulate the view further for mobile, like showing different navigation menus or hiding certain content.

Also font-awesome is installed, so make use of it (http://fortawesome.github.io/Font-Awesome/), add the classes to add the icons…

```
<i class=”fa fa-trash”></i>
```

Style

We are basically writing less files here, then compiling them manually and submitting both to the style folder. If you use SASS, well sorry, I use LESS, thats it really, no other reason. Sure you can use SASS if you wish, but I will only except LESS into the razorCMS published ecosystem. It is silly to support to stylesheet languages so we are going to stick to LESS.

In any theme sent in for publishing on the razorCMS website, I expect to see LESS and compiled CSS for each style file. If you do not know what LESS is, http://lesscss.org/ this should help. It quite simply will transform how you write your style. I suggest you get to know this before going further.

If nothing else, I at least expect to see hierarchical style written, like so:

```
body
{
    background-color: #F4F4F4;

    .template-header
    {
        background-color: #F4F4F4;
        padding-top: 20px;
        padding-bottom: 10px;
        border-bottom: 1px solid #E2E2E2;

        .template-header-menu
        {
            margin-bottom: 20px;
        }
    }
}
```

Where possible, it would be good to put default colours, sizes, borders etc in mixins at the top of the sheet

```
@light-grey-background: #f4f4f4;

body
{
    background-color: @light-grey-background;

    .template-header
    {
        background-color: @light-grey-background;
        padding-top: 20px;
        padding-bottom: 10px;
        border-bottom: 1px solid #E2E2E2;

        .template-header-menu
        {
            margin-bottom: 20px;
        }
    }
}
```

This makes adjusting the theme easier for you and others. Once you harness the power of less you will be knocking out themes extra quick.

So you will be writing LESS and compiling to CSS as you go, we could do this automagically on first load of the system, but I would rather save on the overhead of that, so we will compile manually, I will not sacrifice overhead and loading for lazyness on compiles (we don’t have to do it often so lets get into the habit). You have a few options here, create a task to do this for you (such as grunt) or you can do this the manual way. I use sublime text editor, this comes with several LESS compiler add ons (I use node.js less), simply instal and ctrl + B compiles to css, I think the latest version even minifies and adds a map file.

If you cannot find a less compiler for your text editor of choice, I suggest it is time to change, sublime is a good place to start.

Thats it really, it’s up to you now

I can only open the door for you, now you need to explore what can be achieved. To get the most out of your theme, and your site, look into the resources included above, you have a full scalable, colourable icon set, you have the power of bootstrap, and the beauty of LESS.

Feedback and helping out

If you have any feedback on razorCMS, please don’t hesitate, any feedback is good feedback, seriously, even negative just let me know. I can be contacted through the support forum through the handle smiffy6969 or through the razorCMS main website contact page.

If you want to offer more than feedback, great you’ll make me very happy. I do all this for nothing, you get all my work for absolutely zero cost, yes my wife thinks I’m nuts but that’s not the point. If you want to offer anything, be it time, your expertise, money, livestock, alcohol, moustaches or even munchies I will be your friend for life and more to the point you’ll help me feed my children whilst proving to my wife that open source is worth it.

Signing off

That’s about it for now, if you think this manual is utter pants (I am trying to do 30 tasks at once here), have spotted errors in the manual think you would like to re-write this for me…. Please let me know, someone much more passionate about writing docs is so very welcome to come and put my cards in order.

smiffy6969
www.razorcms.co.uk
ulsmith.net