

WordPress/building themes for designers

It's never been easier to power your website with WordPress. Elliot Jay Stocks explains how themes work, the simple way to edit them, and how to incorporate some basic plug-ins

Knowledge needed Basic XHTML/CSS

Requires Text editor, FTP client, self-hosted WordPress installation

Project time 1 hour

In the last few years WordPress has grown from a simple blogging engine to one of the web's favourite publishing tools. With its growing feature set and the recent improvements found in version 2.5, it's now commonly used as a full-blown CMS. With the wealth of themes available for free on the internet (plus the growing popularity of paid-for 'premium' themes that include advanced, CMS-like functionality), it's never been easier to use WordPress to power your website.

For many designers (including me), the idea of fiddling around with PHP isn't in the least bit appealing. But don't be put off! In this tutorial, aimed specifically at designers, I'll be covering some basics and explaining why it's not that scary to look under the hood. You'll learn about the basic structure of a WordPress theme, how to manipulate the files to add your own customisation, how to incorporate some basic plug-ins, and where to take your skills next in order to create your own theme. We'll even touch on how to use WordPress as a CMS while keeping the PHP to an absolute minimum.

If you're new to WordPress, the first thing to understand is that the look and functionality of your website (whether a simple blog or a fully-fledged product portfolio) is controlled by a 'theme'. Two themes come bundled with the WordPress installation, Classic and Default (also known as K2), but there are thousands of others available online. All work in the same way, so in order to understand WordPress, it's important to grasp how themes work.

Imagine the WordPress core files as your skeleton: they can't change because you need them in order to work. Now imagine the theme as your muscles and skin layer: it sits on top of the skeleton, but its shape and strength vary slightly to determine your main appearance and ability. On

top of that is the CSS file in your theme, and it's best to imagine that as your clothes: a final layer to craft your own personal look.

At its simplest level, a theme is basically a collection of files that work together to make your site function on top of the WordPress engine at its heart. You can change themes very easily through the WordPress admin area. Follow the download and installation instructions found at tinyurl.com/4oc5rv and try swapping between the themes. See what difference it makes? But remember: you're not just swapping CSS files over; you're swapping over *all* of the files in each theme, telling the engine which 'muscles and skin' layer to use, as well as the 'clothes' layer on top of that.

Most of the filenames in a theme directory are self-explanatory, but sometimes confusion can arise regarding the difference between **index.php**, **single.php** and **page.php**. Which file should you edit to exact a certain change?

Index.php is the homepage of your site and will display as many blog posts as you set in the admin. It's effectively the 'main' file in your theme and contains several includes, such as calls to the header, footer and sidebar, which are discussed below. **Single.php** is the file that controls the page that displays a single post, and is especially important if you only display a snippet of the blog posts on the homepage – **single.php** is responsible for displaying the full entry. If you look at its structure, you'll see how similar it is to **index.php**, but you can edit this file to contain different elements to the homepage if you wish. **Page.php** refers only to the template controlling static pages, which are discussed later in this article. If you've ever worked on a site that uses includes to render recurring HTML snippets, you'll see the same thing in play in a WordPress theme. Every main PHP file contains includes such as

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

which tells it to render the contents of the header file, found in the theme as **header.php**. Change something in **header.php** and you'll see it reflected across the entire site. The same goes for **footer.php** and **sidebar.php**.

Starting point

Creating your own theme is extremely simple, and in essence involves you doing nothing more than modifying an existing theme's files. A site's look can be changed dramatically just by editing its CSS file (always called **style.css** in a theme), but changing the content of the PHP files will allow further flexibility and functionality. You can use one of the default themes that come with WordPress as your starting point, or you can use any pre-existing theme you like, but choosing a well-written theme will make your life a lot easier.

There are now some good 'bare bones' themes available, designed specifically for you to build upon. I released 'Starkers' (tinyurl.com/4z86u6) at the beginning of the year to aid the theme-building process; it's basically a stripped-down version of K2 with almost all of the non-semantic HTML removed and a CSS reset applied. 'The Sandbox' is something similar but with a lot more power, and was built from the ground up instead of being based on K2. It presents a wealth of options by automatically generating <body>, post, and comment classnames, which give you a high level of control over the content. Read more about it at plaintxt.org/themes/sandbox. It's also worth checking out Thematic by Ian Stewart (code.google.com/p/thematic), which at the time of writing is under development but nearing fruition.



WordPress.org Head here for the self-hosted version of WordPress and the comprehensive knowledge base. Visit .com for the more 'consumer level' hosted version



Up the revolution The Revolution theme (revolutiontheme.com) is one of many that boasts advanced features, and is a great starting point for you to build your own themes

You can use any pre-existing theme, but a well-written one will make your life a lot easier

Another tip would be to find a theme that contains a particular bit of functionality you want to emulate (like a differently-styled first post, for example, as found in the Revolution theme at revolutiontheme.com/pro/) and use that as a starting point. As you become more experienced at theme editing, you'll be able to just take select blocks of code from a variety of themes as and when you need them.

The loop

A quick scout around the WordPress Codex (codex.wordpress.org) reveals many references to 'the loop', but what is it? In the default template, it looks like this:

```
<?php if (have_posts()) : ?>
<?php while (have_posts()) : the_post(); ?>
<div class="post" id="post-<?php the_ID(); ?>">
<h2><a href="<?php the_permalink() ?>" rel="bookmark" title="Permanent
Link to <?php the_title_attribute(); ?>"><?php the_title(); ?></a></h2>
<small><?php the_time('F jS, Y') ?> <!-- by <?php the_author() ?> --></small>
<div class="entry">
<?php the_content('Read the rest of this entry &raquo;'); ?>
</div>
<p class="postmetadata"><?php the_tags('Tags: ', ' ', '<br />'); ?> Posted in
<?php the_category(' ', ' ') ?> | <?php edit_post_link('Edit', ' ', ' '); ?> <?php
comments_popup_link('No Comments &#187;', '1 Comment &#187;', '%
Comments &#187;'); ?></p>
</div>
<?php endwhile; ?>
<div class="navigation">
<div class="alignleft"><?php next_posts_link('&laquo; Older Entries') ?></div>
<div class="alignright"><?php previous_posts_link('Newer Entries &raquo;')
?></div>
</div>
<?php else : ?>
<h2 class="center">Not Found</h2>
<p class="center">Sorry, but you are looking for something that isn't here.</
p>
<?php include (TEMPLATEPATH . "/searchform.php"); ?>
<?php endif; ?>
```

Scared? Don't be! The loop is simply a bit of code that controls the individual post, and although it only appears once in the PHP file, it gets repeated as

Expert tip Custom fields

One of the most powerful features of WordPress is something you could easily miss if you didn't know about it – the ability to add custom fields to your posts. What this essentially means is that you can add extra fields of your choosing beyond the 'title' and 'body' defaults, and therefore use WordPress more like a standard CMS. Custom fields work by generating key/value pairs.

Let's try adding a background image to our posts on the front page. We'll create a custom field called **background** and use it to pull in photo URLs from Flickr.

In the **Write Post** screen, expand the **Custom Fields** section and write a new key called **background**. You'll only ever need to input this once: it'll appear on the drop-down list on the left from now on. In the **Value** field, enter the URL of a photograph you'd like to use as a background. Now open up **index.php** from your theme directory and just inside the opening code of the loop, add the following lines:

```
// check for background image
$background = get_post_meta($post->ID, 'background', $single = true);
```

Now find the line that dictates the post's ID. Change it to the following (you may have to substitute some HTML elements, depending on how you have your page set up):

```
<?php // if there's a background image
if($background != "") { ?>
< li class="post" id="post-<?php the_ID(); ?>" style="background:
url(<?php echo $background; ?>")">
```

The inline style will then render the background image of the post using the URL you supplied in the **Value** field. This is the same technique we used on carsonified.com. Detailed information can be found at codex.wordpress.org/Using_Custom_Fields.

many times as necessary when the page is rendered in the browser. If you've set your site to display 10 posts per page, it'll do just that with you only having to worry about having *one* loop in your PHP file.

Now it's time to see just how easy theme editing can be. Assuming you have WordPress installed and you're viewing your brand new WordPress-powered site in a browser, we're going to edit a small amount of code so you can see how the page changes; just make sure you refresh your browser after you make changes to the code. Let's look at rewording the metadata. Find the code

```
Posted in <?php the_category(' ', ' ') ?>
```

located in the **metadata** paragraph, and change it to

```
Filed under: <?php the_category(' ', ' ') ?>
```

Save your text file (if you're working directly from your FTP app) and refresh your browser. Notice the difference in the text? Let's edit the text a bit more, but this time relating to the part of the code that displays how many comments the post has. On the same line, find the code

```
<?php comments_popup_link('No Comments &#187;', '1 Comment &#187;',
'% Comments &#187;'); ?>
```

and change it to

```
<?php comments_popup_link('No-one has commented - be the first!', '1
lonely comment', '% Comments'); ?>
```

Now you see how easy it is to manipulate the text in the loop. Although this is only simple, the same principle applies for advanced code editing, where you can start changing bits of PHP to get your desired effect.

Part of the appeal of WordPress lies in its ability to integrate third-party plug-ins. Like software plug-ins, those for WordPress provide



advanced functionality, with no need for you to write advanced code. After installation (where you simply upload the plug-ins to the correct folder on your server and 'activate' them through the admin panel), most only require you to paste one line of code in whichever place you'd like the plug-in's output to appear. If you're looking for plug-ins, a great place to start is wordpress.org/extend/plugins/.

Let's integrate a plug-in into our site so we can display photographs from a Flickr account in the header. There are a number of plug-ins available to do this job, but here we'll be using Dave Kellam's FlickrRSS. This can be found at eightface.com/wordpress/flickrRSS/.

Firstly, download FlickrRSS and follow the installation instructions to upload the files to the **wp-content>plugins** directory. All plug-ins need to be activated before they can work, so log in to your WordPress admin area and click on the plug-ins link on the right. Hit the **Activate** button and the plug-in will be ready for set-up. If you then visit the **Settings>FlickrRSS** in your admin panel, you can configure a few basic options about how the data is rendered to the outputted page.

With the set-up complete, all we need to do is edit the **header.php** file with our text editor and add the line

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

Put this just above the closing tag of the div with an id of **header**. Refresh your browser and you should see your Flickr images appearing underneath the site description. You can now style these as desired by editing the **style.css** file in your theme's directory.

Posts v pages

There can be some confusion surrounding the difference between posts and pages, because essentially they share the same architecture. However, pages differ from posts because they act like 'static' pages on a site (like 'about', 'contact', etc), sitting outside of the regular blog chronology. Importantly, they sit outside of the normal permalink structure (defined in **Settings>Permalinks**) and instead have URLs like <http://example.com/info/>.

In a standard theme directory, there is a file that controls the structure of a page: the aptly-named **page.php**, mentioned previously. You might want to include different elements on your static pages to the blog post pages, or visually differentiate them. What many people don't realise is that you can create your own page templates so that each page your viewer sees has a slightly different look.

To do this, first open **page.php** in your text editor, save it as **page-info.php** and upload this new file to your theme directory. Make sure it has some kind of recognisable difference to the normal page template.

Now go into your WordPress admin area and choose **Write>New Page**. In the options below the main text fields, find the **Page Template** menu and you'll see a drop-down box containing the available templates. At the

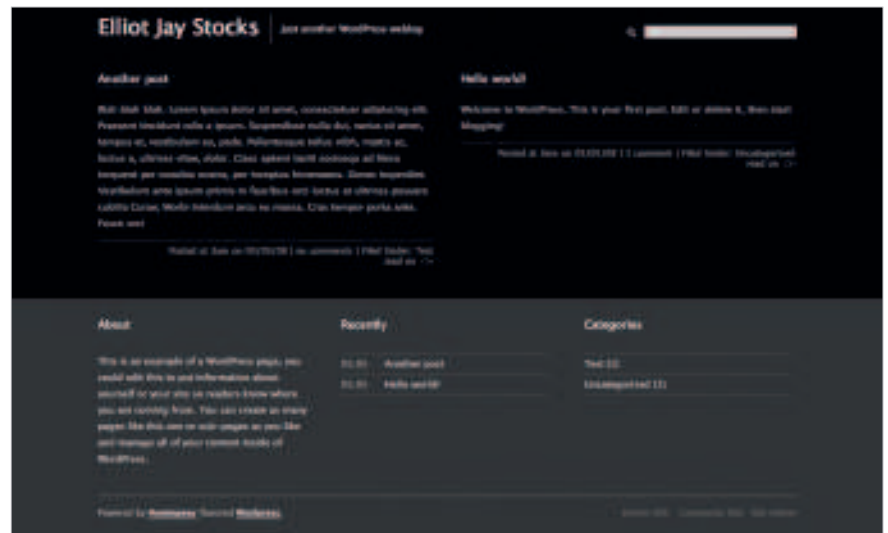
Resources Where to learn more



WordPress Codex
The main port of call for those starting out is the official WordPress Codex, which has information on everything from installing WordPress to file and plug-in management, plus a support forum, a troubleshooting guide and a handy FAQ section. codex.wordpress.org



Secrets of WP theming
The WordPress community is very active so it's very easy to find information online, such as this article by Chris J Davis, which nicely outlines some of the lesser-known features of the Theme System in WordPress 1.5. www.chrisjdavis.org/secretsof-wp-theming-part-1



Inventive Hemingway (warpshire.com/hemingway/) demonstrates different functionality to the default themes; it's worth looking 'under the hood' at its **functions.php** file

moment, ours doesn't appear on the list, so in order to do that, we need to add the following code at the top of our PHP file:

```
<?php
/*
Template Name: Info
*/
?>
```

If you refresh the admin screen in your browser and locate the drop-down menu once more, our new template's name (**Info**) will be in the list. Select this file, publish the page, and view the 'info' page of the site in your browser.

WordPress also enables you to define any page as the homepage (in **Settings>Reading**), and while this seems like a small feature, it's incredibly powerful. It means you can push the 'blog' part of your site back if it's not the main feature. Perhaps you might want 'about' to be the index page your users see, or some kind of splash page that incorporates snippets from other pages and blog posts? This can all be done with very little hassle.

We've only scratched the surface of the theme creation and customisation process, yet you're probably already seeing the potential that WordPress offers. You've probably encountered various options and extras in the admin section that you want to explore, and you've hopefully been inspired by the capabilities offered by some of the great themes and plug-ins that exist in the WordPress community.

Have a go at creating your own theme: simply pick a theme you like as a starting point and gradually change parts of the code until you have an understanding of the process. When you're ready to go live with your new theme, why not consider releasing it back to the WordPress community for free? ●

Pick a theme you like as a starting point and gradually change parts of the code



About the author

Name: Elliot Jay Stocks
Site: elliotjaystocks.com
Areas of expertise: Design, CSS, web standards, Photoshop
Clients: Blue Flavor, WordPress, The Beatles
What makes you angry?: Unquestioning ignorance