

WordPress Intensive: From blogging to building dynamic web sites

Handout Addendum

David Tamés

<http://MediaTechTonic.org>

This handout contains additional information based on questions and issues discussed during the session. Both this and the original handout document are available online at <http://MediaTechTonic.org>.

Two approaches for developing custom themes

There are two approaches for developing a custom WordPress theme.

1. You can start with an existing theme you like and edit the template files and stylesheet as needed, or
2. You can start with a model of your site developed using HTML and CSS and then add template tags and loops as needed. It helps to use a user interface framework like the Yahoo Interface Library to start with a foundation with good cross-browser compatibility. Dealing with the obnoxious (some would say intentional) bugs in Internet Explorer should be avoided whenever possible.

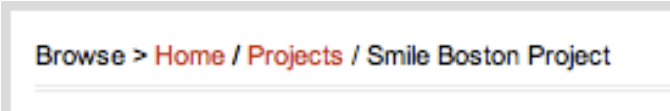
While option 1 is easier, option 2 provides complete flexibility. In any case, I suggest studying the template files in the default themes and one or two other themes to get familiar with how things work. There's excellent documentation on this in the WordPress Codex.

A pair of code fragments

Here's couple of code fragments you might find useful when developing a custom theme or tweaking an existing theme. Looking at existing themes that do interesting things is a great way to learn the ins and outs of custom WordPress themes and template tweaking.

Breadcrumb navigation

With the **Breadcrumb Navigation XT** plug-in you can add breadcrumb navigation to your posts. After installing and activating the plug-in, you will need to add something along the lines of the following to your template files (results on the right):



Browse > Home / Projects / Smile Boston Project

```
<div class="breadcrumb">
  <?php
    if (class_exists('breadcrumb_navigation_xt')) {
      echo 'Browse > ';
      // New breadcrumb object
      $bco = new breadcrumb_navigation_xt;
      // Options for breadcrumb_navigation_xt
      $bco->opt['title_blog'] = 'Home';
      $bco->opt['separator'] = ' / ';
      $bco->opt['singleblogpost_category_display'] = true;
      // Display the breadcrumb
      $bco->display();
    } ?>
</div> <!-- /breadcrumb -->
```

List of sticky posts

You can get anything you need from WordPress using the **WP_Query** function. The **Post-Thumb Revisited** plug in used for creating thumbnails in this example. Put each block of content in it's own div. I don't use it in this example, but another useful function is **get_post_meta**, which allows you to access metadata stored in Custom Fields (see the WordPress Codex for details). One possible use of Custom Fields would be to store the date on which a story is featured on the home page.

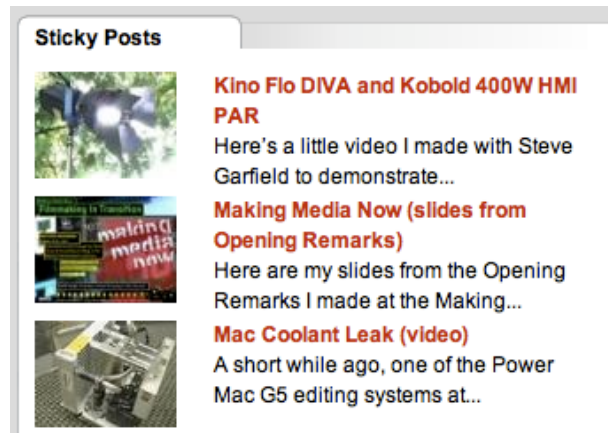
```
<div class="thumb-list">
<h2>Sticky Posts</h2>
<!-- 42 = Sticky category id, site dependent -->
<?php $recent = new WP_Query("cat=42&showposts=8");
while($recent->have_posts()) : $recent->the_post();?>
    <div class="thumbleft">
    <!-- requires Post Thumb Revisited plug-in, p=link to post -->
    <?php the_thumb('LINK=p'); ?> </div>
    <div class="thumbright">
    <h3><a href="<?php the_permalink() ?>" rel="bookmark">
    <?php the_title(); ?></a></h3>
    <!-- show a little bit of the content -->
    <?php the_content_limit(60, ""); ?>
    </div>
    <div style="clear:both;"></div>
    <?php endwhile; ?>
</div> <!-- /thumb-list -->
```

The use of divs and common HTML tags allows you do to formatting through the stylesheet, that way the HTML in the templates is very simple. This also makes it easier to tweak the look and feel without having to go into the template files. The related CSS that defines thumb-list, thumbright and thumbleft is below, along with what the section of the page looks like, styling for the header, headings, and content is defined elsewhere in the stylesheet.

```
.thumb-list {
    background: #FFFFFF;
    url(images/thumb-list.gif);
    float: right;
    width: 316px;
    margin: 0px 0px 10px 0px;
    padding: 5px 10px 0px 10px;
}

.thumbleft {
    float: left;
    width: 100px;
}

.thumbright {
    float: right;
    width: 215px;
}
```



Creating your own page templates

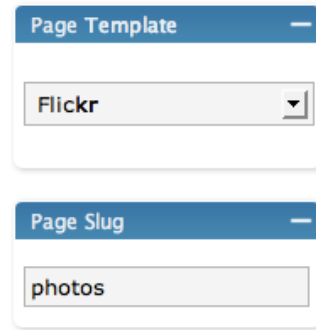
To create a custom page you must first create a new Page Template for that page. Let's say you want a special page that displays images from Flickr, so we'll call it flickr.php. In the theme directory, you identify custom templates to WordPress with the the following at the top of the page:

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

The above code lets WordPress know that flickr.php is the "Flickr" template. This name will appear in the Page Template pop-up as one of the options. Avoid using names of common or existing template files.

Template files must be named with a .php extension. Once you create your new Page Template, you'll be able to choose your custom Page Template when editing a page (illustrated on the right).

Here's an example of a custom Page Template that uses PHPFlickr to display "interesting" and "recent" images from a Flickr photostream. You will need a Flickr API Key, for information on that visit http://www.flickr.com/services/api/misc.api_keys.html



```
<?php
/*
Template Name: Flickr
*/
?>
<?php get_header(); ?>
<div id="content">
<div id="contentleft">
<div class="postarea">

<!-- display photos from Flickr -->
<h1>Photos from Flickr</h1>
<p> Visit my <a href="[insert-link-to-your-flickr-page-here]" title="Link:
kino-eye's photo page on flickr">photo page</a> on Flickr.</p>
<?php require_once("[insert-path-to-phpflickr-here]/phpFlickr.php");
// Create new phpFlickr object
// To get a Flickr API Key go to
//
$f = new phpFlickr("[insert-flickr-api-key-here]");
$f->enableCache("db", "mysql://[insert-mysql-db-path-here]");
$i = 0;
$person = $f->people_findByUsername('kino-eye');
// Get the friendly URL of the user's photos
$photos_url = $f->urls_getUserPhotos($person['id']); ?>

<!-- interesting photos -->
<h4>My most interesting photos (according to Flickr):</h4>
<p> <?php // Search for most interesting by user_id
$photos_mine = $f->photos_search(array("user_id"=>$person['id'],
"sort"=>"interestingness_desc", "per_page"=>35));
$i = 0; foreach ($photos_mine['photo'] as $photo) {
// Build image and link tags for each photo
echo "<a href=http://www.flickr.com/photos/$photo[owner]/$photo[id]>";
echo "";
echo "</a>";
$i++;
} ?> </p>
<div style="clear:both;"></div> <p> &nbsp; </p>

<!-- Recently added photos -->
<h4>Photos I've recently added to Flickr:</h4>
<p> <?php // Get the user's first 36 public photos
$photos = $f->people_getPublicPhotos($person['id'], NULL, 35);
// Loop through the photos and output the html
$i = 0;
foreach ($photos['photo'] as $photo) {
echo "<a href=$photos_url$photo[id]>";
echo "";
    echo "</a>"; $i++;
} ?> </p>
<div style="clear:both;"></div>
<p>This page made with <a href="http://phpflickr.com/">phpFlickr<a>.</p>
</div> <!-- /postarea -->
</div> <!-- /contentleft -->
<?php include(TEMPLATEPATH."/sidebar.php");?>
</div> <!-- /content -->
<?php get_footer(); ?>

```

You can see what it looks like at <http://kino-eye.com/photos/>

Anatomy of a WordPress theme

Beyond the minimal set of required template files (in bold), most themes define more. Here's a listing of the files typically defined in a theme.

404.php	When a server error 404 occurs ("Page not found") this is the template used.
archive.php	The archive template. Used when a category, author, or date is queried. If the category.php, author.php, and date.php templates exist, they will be used instead of archive.php respectively.
category.php	The category template.
comments.php	The comments template. If this file does not exist, wp-comments.php is used instead.
date.php	The date template. Used when a WordPress needs to display posts by date.
footer.php	This template defines what's in the footer. This file must exist.
functions.php	If you've written functions that are shared among the various template files, they should be defined here.
header.php	This template defines what goes in the header. This file must exist.
index.php	The primary template. This file must exist.
links.php	The template used for listing links.
page.php	The single page template. This is the template used when WordPress displays a single page. If this template does not exist, WordPress will use index.php instead.
sidebar.php	The template that defines the sidebar. This file must exist.
screenshot.png	This image is displayed in the administration control panel in the list of themes.
search.php	The template used for search results.
searchform.php	The form to be used for searching.
single.php	The single post template. This is the template used when WordPress displays a single post. If this template does not exist, WordPress will use index.php instead.
style.css	The stylesheet for the theme. This file must exist.
images	A folder in the theme directory holding theme-specific images.

Creating a custom home page

Sometimes you don't want the default 'blog' home page, you might want to take complete control of how the home page looks, and make it completely different from the blogging portion of the site. You can have a specific page with your own custom template be the home page for the site. Setting up a static page as the home page is described in the article "Having a static page at the front of the blog" at <http://faq.wordpress.com/2007/02/20/having-a-static-page-at-the-front-of-the-blog/>

Here is the list of theme files recognized by WordPress. Of course, your theme can contain any templates, stylesheets, images, or other files you wish to include. Just keep in mind that the following have special meaning to WordPress.

Code libraries

These code libraries are referenced in this handout.

PHPFlickr

<http://phpflickr.com>

A PHP code library you can use for adding Flickr images to page templates.

Yahoo! Interface Library

<http://developer.yahoo.com/yui/>

If you want to create a theme from scratch that is highly cross-browser compatible, you may want to consider basing your design on an interface library like this which allows you to start with a baseline of clean, cross-browser compatible HTML and CSS. Lots of UI goodies and examples.

Carp

<http://carp.docs.geckotribe.com/>

If you want to do interesting things like aggregate, filter, and display RSS feeds on your site, Carp (Caching RSS Parser) provides a nice way to do it. For example, the "Links" page template on my blog uses Carp to cache the RSS feed from my del.icio.us/cinemakinoeye/kinoeyepicks feed, the page is at: <http://kino-eye.com/links/>

WordPress plug-ins

These WordPress plug-ins are referenced in this handout.

Post Thumb Revisited

<http://www.alakhnor.com/post-thumb/>

Scan posts for images, it can show a thumbnail linked to the post or show thumbnails from the most recent posts or thumbnails from random posts. Thumbnails are dynamically created when needed and then saved, no complex thumbnail management.

Breadcrumb Navigation XT

<http://mtekk.weblogs.us/code/breadcrumb-nav-xt/>

Generates breadcrumb navigation bars for your WordPress blog.

Further reading

WordPress Codex

<http://codex.wordpress.org>

WordPress Lessons

http://codex.wordpress.org/WordPress_Lessons

WordPress Themes

http://codex.wordpress.org/Theme_Development

WordPress Themes: Starting Your Custom Theme (straightforward step-by-step tutorial)

<http://websavvymama.com/blog/2007/wordpress-themes-starting-your-custom-theme/>

So you want to create WordPress themes huh? (detailed Windows focused step-by-step tutorial)

<http://www.wpdesigner.com/2007/02/19/so-you-want-to-create-wordpress-themes-huh/>