



Lesson #14

Developing single.php and comments.php

By

Lynette Chandler
Tech Based Training

<http://TechBasedTraining.com/>

**NOTICE: You Do NOT Have the Right
to Reprint or Resell this Report!**

You Also MAY NOT Give Away,

Sell or Share the Content Herein

If you obtained this report from anywhere other than **Tech Based Training.com** , you have a pirated copy.

Help stop internet piracy by letting me know. Send email to customerservice@techbasedsupport.com

© 2010 Copyright Lynette Chandler

Legal Disclaimer: Keeping things simple here, I have to say that I cannot promise you success. I can give you direction and advice based on my experiences and good internet marketing practices. What you do with this information is up to you. As a Tech Based Training member you agree to not hold me responsible for your results.

Just Two More

Can you believe we only have two more template files to develop – single.php and comments.php – and our theme will be fully functional? No, we aren't at the end of building our theme yet. Far from it. As a matter of fact, it is quite impossible to finish a theme because each one of us will have a different idea what is finished depending on how you will ultimately use the theme.

There will always be little things you can add here and there to improve your theme. Many are not possible for you to build until you know the web site you will apply the theme to.

That is why we will tackle those areas – the custom areas in our later lessons. Look at it in phases. Broadly, there are three phases. We are still in the first phase, where we convert straight HTML into WordPress and make it fully functional. Getting the structure and the foundation set.

The second phase is to flesh out the theme even more using optional template files that WordPress is able to recognize like archive.php, page.php and author.php. A theme will absolutely function without these templates. They only serve to enhance or make the theme more useful.

The third phase is writing code, looking at certain case scenarios, what and how to do each of them.

Without further ado, let's get this template file built. As usual, gather your tools. Launch PsPad, if you used Uniform Server, start it. Launch Firefox, bring up your Sandbox blog in the browser.

Create single.php

The template file single.php is the file that is used to display each individual post. In a typical WordPress blog, you land on the home page, you'll see a list of posts. That is displayed using the template file index.php. When you click on a post you want to read, you are most likely taken to another page where only that post is

displayed. None other. This is displayed using the single.php template file. For all practical purposes, WordPress can function without single.php. When you review your Sandbox blog now with the template we have built so far, you will see that it does show the post on its own page.

But there is a problem. There are no comments and no comment form. Also, the big images and headers are still on the top. Something that is best only on the home page to draw people in to other content. On a single post page, you want them to concentrate on your content.

To make this happen you have two choices. Further build up index.php by using conditional tags or create single.php. Conditional tags are tags that only work if a certain condition is met. For example. If reader is on a single post page, then display this, don't display that. However, I've found creating single.php is simply cleaner, easier to work with and less confusing.

Trust me, the more conditions you put on a template, the more confusing you will get. I recently used a lot of conditional tags on a client's site which happens to be quite complex – I regret not using single.php now because it takes me longer to troubleshoot and just difficult to navigate in the code. On screen to the client and visitors it looks great but inside, its a jungle. Separate templates where possible will give you less headaches.

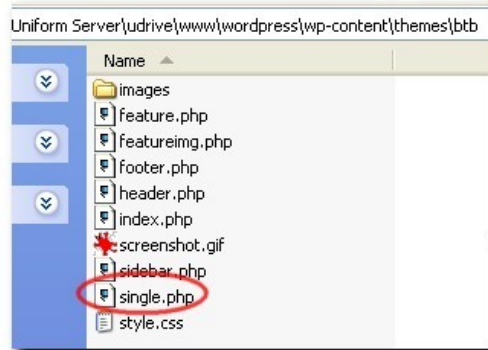
This is not to say don't use conditional tags. There will be opportunities when using conditional tags are probably the best thing to do and sometimes you have no choice. That will be your call. We will cover conditional tags more in phase three of this theme building process so let's not get ahead of ourselves.

We've already established we should create single.php. We won't be building this template from scratch as it is not necessary and a waste of time to do that. We will simply clone index.php then make modifications to it.

Here's how we do it. In PsPad, open your Sandbox blog's index.php file. Once you have the file opened, click File >> Save As or hit F12. A save as dialog box appears.

In the file name box, replace index.php with single.php. Click Save.

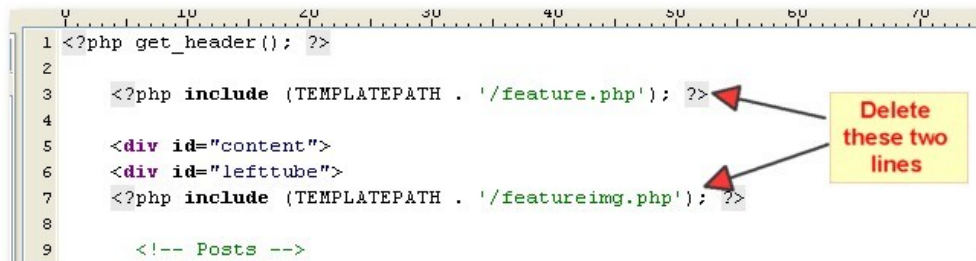
If you navigate to your theme folder, you should now see a new file single.php.



Remove unwanted elements

We've already briefly discussed that we do not want the featured content area and images on single post pages. So the first thing we do is get rid of unwanted elements.

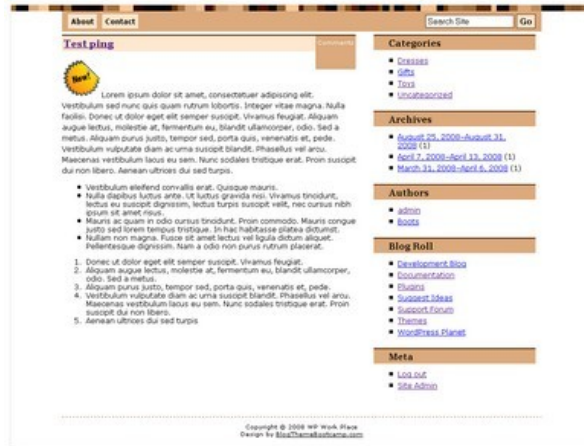
In PsPad, with single.php still open, delete the feature.php and featureimg.php include codes.



In your browser, click on a post in your Sandbox blog if you haven't done so already. The page should only display the single post without featured images on the top.

You may also notice that the sign up box has disappeared as well. That's because the sign up box was originally included in the featured content area. If a sign up

box is desired on all pages, we will later add this back inside the sidebar using conditional tags. This is what my single post page looks like.



Fixing up

A few things may seem amiss. One thing I've noticed on my theme is the number of comments do not display anymore. This is something that I see on certain installations where the code works on the home page but not other pages. Happily, we have another code we can use to bring that back.

If your number of comments is missing as well, replace this

```
14 <div class="commentnbr">
15 <h4><?php comments_popup_link('0', '1', '% '); ?></h4>
16 <p>Comments</p>
17 </div>
```

with this line of code.

```
<?php comments_number('0', '1', '% ');?>
```

```
10 <div class="posttitle">
11 <div class="commentnbr">
12 <h4><?php comments_number('0', '1', '% ');?></h4>
13 <p>Comments</p>
```

The design may also seem a bit strange now that the images are gone. It could be we may be used to seeing the images there after so long. We will not be tweaking that to keep things simple. If you have strong CSS skills and have worked with WordPress before, you can go ahead and edit it on your own.

Adding author information

In most themes, the author's name would appear almost everywhere. We did not add the author's name to index.php so we will add it here since single post pages are supposed to contain more information.

Around line 15, right after the title of the post, I'm going to press enter to make a new line then paste this code there.

```
by <?php the_author(); ?>
```

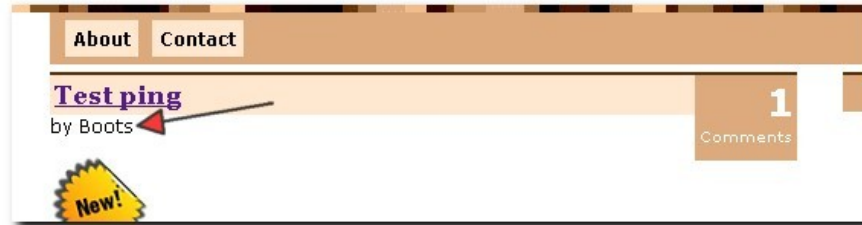
This:

```
14 </div>
15 <h1><a href="<?php the_permalink() ?>"><?php the_title(); ?></a></h1>
16
```

Becomes this:

```
15 <h1><a href="<?php the_permalink() ?>"><?php the_title(); ?></a></h1>
16 by <?php the_author(); ?>
17 </div>
18 <?php the_content(); ?>
```

The result upon refreshing my browser is this:



Instead of the `_author()`; you can also use this tag the `_author_posts_link()`; . The first which we used in our example only displays the author's name. The second displays the author's name and links it to a list of the author's posts. This is very useful if the blog is maintained by multiple authors.

Adding the date

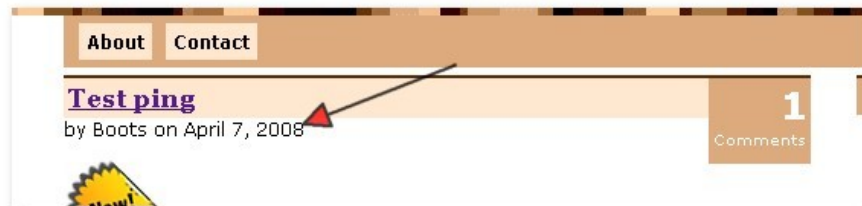
Like the author name, we did not include this in `index.php` so we will add them now. I'm going to put the date right after the author's name. Immediately after the code we just pasted around line 16, I'm going to add this code.

```
on <?php the_date(); ?>
```

So now my code looks like this:

```
15 <h1><a href="<?php the_permalink() ?>"><?php the_title() ?></a></h1>
16 by <?php the_author(); ?> on <?php the_date(); ?>
17 </div>
```

With a result like this:



Date and author name information is optional. Most themes have it on `index.php` too. In fact, if you wish you can add them on your own. You can even add other

bits of information like Posted in which category by using the `the_category` tag. Find out more about that tag [here](#). I've left it out to keep the theme as simple as possible.

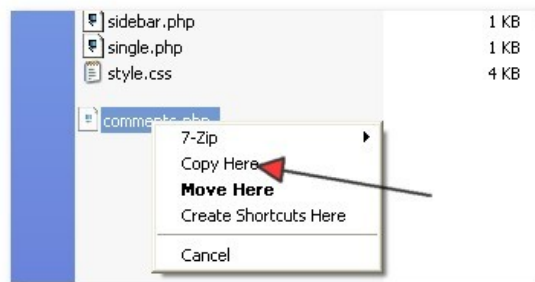
Adding comments

Now it's time to add comments. The comments template is pretty complex. Unlike the other template files we have build so far, this one consists of a lot of php coding and is not very much dependent on the theme. For that matter, my normal approach is to copy the `comments.php` template from the default theme into the theme I'm building. This cuts down development time considerably and is simply a way of smartly re-using code.

Just one caveat. If you copy the code from another theme, make sure that you have the rights to do so. This is why I normally use the default theme `comments.php` because it is an open source license which allows me to freely adapt, copy and re-use.

If your Sandbox is on your own computer, copying a file couldn't be easier. Simply open two instances of Windows File Explorer, navigate one to the default theme and one to the Blog Theme Bootcamp theme. Right click on `comments.php` in the default theme, drag and drop it over at the Blog Theme Bootcamp theme folder.

A small dialog box will pop up. Click copy here.



If your Sandbox is on the Internet, use cPanel File Manager to do it. Depending on your cPanel version this may look different. If it is different or you run into

problems, please seek help from your web host first. This is something they can certainly help you with.

Web hosts differ greatly so there is a good chance they may have a different cPanel theme that does not behave like outlined below. Or they may have some custom built solutions or a different version of cPanel.

Log in to your cPanel. Click on File Manager



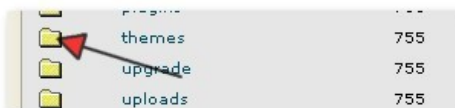
Navigate to the theme folder by clicking once on the folder icons to open the folder. First, open public_html



then, wp-content folder – if you installed WordPress inside a folder you may have to click to open that folder first before you find wp-content.



Then themes



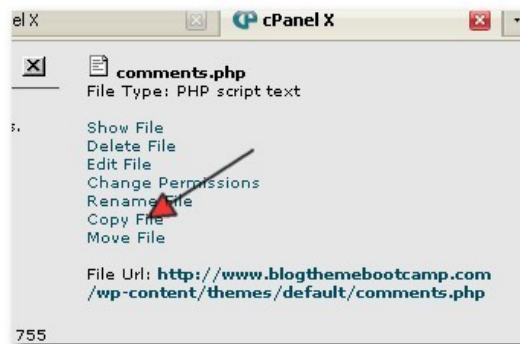
Then default



Find comments.php, click on the file name this time (not the icon) to select the file.



Once you click on it, a bunch of links appear on the top right corner of the screen. Click Copy File



The screen to your left changes once more. Like the above click on the icons for public_html >> wp-content >> themes. Stop right there. Find your Blog Theme Bootcamp theme folder. Click on the name of the folder (not the icon) to select. The copy process will begin. When successful, you will see something like this on the top right of your screen.



Once you have successfully copied the comments.php file into your Blog Theme Bootcamp folder, return to Pspad. If you don't have your single.php file still open, re-open it now.

Just under this code the `_content()`; press enter twice to make new lines. Paste the following code.

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

The result should look like this.

```
15 <h1><a href="<?php the_permalink() ?>  
16 </div>  
17 <?php the_content(); ?>  
18  
19 <?php comments_template(); ?>  
20 </div>
```

Refresh your Sandbox blog in your browser – when you are on a single post page. You should see the comments (if any) including the comment submission form.



There is one problem. The comment box is way too large. Don't worry. We will fix that now.

Fixing the comment box

Using Firebug, click Inspect and then click on the comment box. It should show you this line of code that's sizing the comment box.



The rows and columns are hard coded and cols="100" is obviously too wide. We will fix this using CSS so it is more dynamic and not so rigid.

In Pspad, open comments.php. Find the textarea code as you saw in Firebug. It should be around line 94. Delete rows="10" and cols="100". Replace with this code.

```
style="width: 100%; height: 200px;"
```

This will make the box only as wide as the post and the height is just something I thought is a comfortable size you can change the height if you wish. This is how my code looks like now.

```
93
94 <p><textarea name="comment" id="comment" style="width: 100%; height:
↳ 200px;" tabindex="4"></textarea></p>
95
96 <p><input name="submit" type="submit" id="submit" tabindex="5" value="
↳ Submit Comment" />
```

Refresh your Sandbox blog and now the comments box should sit nicely inside the left column.



Complete of phase one

That's it. We now have a fully functional theme. There are no missing pages, as you visit each link and archive. It works. Is it done? Not quite. But we have successfully finished phase one of our theme building. The theme now works on a functional level. You have converted an HTML template into WordPress. You should reward yourself. Good work!

[Here's the theme package](#) for all work we've done from the beginning up to this lesson. If you run into any problems or challenges, compare this and yours to see where the problem may have been. It's usually something small and an easy mistake to make.

Next Lesson...

We're going to take another short detour to talk about Versioning your files as you work. This is a good practice when you're editing existing theme files , a sort of back up technique.

Lynette Chandler

Lynette Chandler

Tech Based Training