WordPress Lessons

WordPress for Beginners

- (01) Introduction to Blogging
- (02) First Steps With WordPress
- (03) WordPress Semantics - Learning the Jargon
- (04) New To WordPress - Where to Start
- WordPress Blog Design and Layout
- (05) Using WordPress Themes
- (06) Theme Development
- (07) FAQ Themes, Layout and Design
- (08) HTML to XHTML
- (09) Business Blogging and Shopping

- (10) WordPress CSS Information and Techniques
- (11) Finding Your CSS Styles
- (12) Creating Individual Pages
- (13) Stepping Into Templates
- (14) Stepping Into Template Tags
- (15) Using Images
- (16) Finding WordPress Help
- (17) I Make Changes and Nothing Happens
- Installing WordPress-An Audio-Visual Presentation (not in PDF)

Note: This compilation was created by JoeTaxpayer who writes on financial topics at JoeTaxpayer.com
It is a PDF snapshot of the WordPress.org lessons as of 6/19/2009. I am not the author and take no credit for this content, I assembled this for my own convenience to send to a printer. I like hard copies. I can write on it, make notes and little smiley faces. If you find a section has been updated, please visit my blog, and drop a note on the "about" page. This is only the section "WordPress for Beginners." If I get comments that this is a good thing, I may do the rest. If I am criticized, I'll take it down, but I've seen a few people ask for the lesson content in this form, so here it is.
Have a nice day.
Joe

Codex

Introduction to Blogging

Contents

What is a "blog"?

"Blog" is an abbreviated version of "weblog," which is a term used to describe web sites that maintain an ongoing chronicle of information. A blog is a frequently updated, personal website featuring diary-type commentary and links to articles on other Web sites. Blogs range from the personal to the political, and can focus on one narrow subject or a whole range of subjects.

Many blogs focus on a **particular** topic, such as web design, home staging, sports, or mobile technology. Some are more eclectic, presenting links to all types of other sites. And others are more like personal journals, presenting the author's daily life and thoughts.

Generally speaking (although there are exceptions), blogs tend to have a few things in common:

- A main content area with articles listed chronologically, newest on top. Often, the articles are organized into categories.
- An archive of older articles.
- A way for people to leave comments about the articles.
- A list of links to other related sites, sometimes called a "blogroll".
- One or more "feeds" like RSS, Atom or RDF files.

Some blogs may have additional features beyond these. Watch this short video for a simple explanation for what a blog is.

The Blog Content

Content is the raison d'être for any web site. Retail sites feature a catalog of products. University sites contain information about their campuses, curriculum, and faculty. News sites show the latest news stories. For a personal blog, you might have a bunch of observations, or reviews. Without some sort of updated content, there is little reason to visit a web site more than once.

On a blog, the content consists of articles (also sometimes called "posts" or "entries") that the author(s) writes. Yes, some blogs have multiple authors, each writing his/her own articles. Typically, blog authors compose their articles in a web-based interface, built into the blogging system itself. Some blogging systems also support the ability to use stand-alone "weblog client" software, which allows authors to write articles offline and upload them at a later time.

Comments

Want an interactive website? Wouldn't it be nice if the readers of a website could leave comments, tips or impressions about the site or a specific article? With blogs, they can! Posting comments is one of the most exciting features of blogs.

Most blogs have a method to allow visitors to leave comments. There are also nifty ways for authors of other blogs to leave comments without even visiting the blog! Called "pingbacks" or "trackbacks", they can inform other bloggers whenever they cite an article from another site in their own articles. All this ensures that online conversations can be maintained painlessly among various site users and websites.

The Difference Between a Blog and CMS?

Software that provides a method of managing your website is commonly called a CMS or "Content Management System". Many blogging software programs are considered a specific type of CMS. They provide the features required to create and maintain a blog, and can make publishing on the internet as simple as writing an article, giving it a title, and organizing it under (one or more) categories. While some CMS programs offer vast and sophisticated features, a basic blogging tool provides an interface where you can work in an easy and, to some degree, intuitive manner while it handles the logistics involved in making your composition presentable and publicly available. In other words, you get to focus on what you want to write, and the blogging tool takes care of the rest of the site management.

WordPress is one such advanced blogging tool and it provides a rich set of features. Through its Administration Panels, you can set options for the behavior and presentation of your weblog. Via these Administration Panels, you can easily compose a blog post, push a button, and be published on the internet, instantly! WordPress goes to great pains to see that your blog posts look good, the text looks beautiful, and the html code it generates conforms to web standards.

If you're just starting out, read Getting Started with WordPress, which contains information on how to get WordPress set up quickly and effectively, as well as information on performing basic tasks within WordPress, like creating new posts or editing existing ones.

Things Bloggers Need to Know

In addition to understanding how your specific blogging software works, such as WordPress, there are some terms and concepts you need to know.

Archives

A blog is also a good way to keep track of articles on a site. A lot of blogs feature an archive based on dates (like a monthly or yearly archive). The front page of a blog may feature a calendar of dates linked to daily archives. Archives can also be based on categories featuring all the articles related to a specific category.

It does not stop there; you can also archive your posts by author or alphabetically. The possibilities are endless. This ability to organize and present articles in a composed fashion is much of what makes blogging a popular personal publishing tool.

Feeds

A Feed is a function of special software that allows "Feedreaders" to access a site automatically looking for new content and then post updates about that new content to another site. This provides a way for users to keep up with the latest and hottest information posted on different blogging sites. Some Feeds include RSS (alternately defined as "Rich Site Summary" or "Really Simple Syndication"), Atom or RDF files. Dave Shea, author of the web design weblog Mezzoblue has written a comprehensive summary of feeds.

Blogrolls

A blogroll is a list, sometimes categorized, of links to webpages the author of a blog finds worthwhile or interesting. The links in a blogroll are usually to other blogs with similar interests. The blogroll is often in a "sidebar" on the page or featured as a dedicated separate web page. BlogRolling and blo.gs are two websites that provide some interesting functions or help related to blogrolls. These sites provide methods for users to maintain these rolls effortlessly and integrate them into weblogs. WordPress has a built-in Link Manager so users do not have to depend on a third party for creating and managing their blogroll.

Syndication

A feed is a machine readable (usually XML) content publication that is updated regularly. Many weblogs publish a feed (usually RSS, but also possibly Atom and RDF and so on, as described above). There are tools out there that call themselves "feedreaders". What they do is they keep checking specified blogs to see if they have been updated, and when the blogs are updated, they display the new post, and a link to it, with an excerpt (or the whole contents) of the post. Each feed contains items that are published over time. When checking a feed, the feedreader is actually looking for new items. New items are automatically discovered and downloaded for you to read. Just so you don't have to visit all the blogs you are interested in. All you have to do with these feedreaders is to add the link to the RSS feed of all the blogs you are interested in. The feedreader will then inform you when any of the blogs have new posts in them. Most blogs have these "Syndication" feeds available for the readers to use.

Managing Comments

One of the most exciting features of blogging tools are the comments. This highly interactive feature allows users to comment upon article posts and link to your posts and comment on and recommend them. These are known as **trackbacks** and **pingbacks** . We'll also discuss how to moderate and manage comments and how to deal with the annoying trend in "comment spam", when unwanted comments are posted to your blog.

- Trackbacks
- Pingbacks
- Verifying Pingbacks and Trackbacks
- Comment Moderation
- Comment Spam

Trackbacks

Trackbacks were originally developed by SixApart, creators of the MovableType blog package. SixApart has a good introduction to trackbacks:

> In a nutshell, TrackBack was designed to provide a method of notification between websites: it is a method of person A saying to person B, "This is something you may be interested in." To do that, person A sends a TrackBack ping to person B.

A better explanation is this:

- Person A writes something on their blog.
- Person B wants to comment on Person A's blog, but wants her own readers to see what she had to say, and be able to comment on her own blog
- Person B posts on her own blog and sends a trackback to Person A's blog
- Person A's blog receives the trackback, and displays it as a comment to the original post. This comment contains a link to Person B's post

The idea here is that more people are introduced to the conversation (both Person A's and Person B's readers can follow links to the other's post), and that there is a level of authenticity to the trackback comments because they originated from another weblog. Unfortunately, there is no actual verification performed on the incoming trackback, and indeed they can even be faked.

Most trackbacks send to Person A only a small portion (called an "excerpt") of what Person B had to say. This is meant to act as a "teaser", letting Person A (and his readers) see some of what Person B had to say, and encouraging them all to click over to Person B's site to read the rest (and possibly comment).

Person B's trackback to Person A's blog generally gets posted along with all the comments. This means that Person A can edit the contents of the trackback on his own server, which means that the whole idea of "authenticity" isn't really solved.(*Note: Person A can only edit the contents of the trackback on his own site. He cannot edit the post on Person B's site that sent the trackback.*)

SixApart has published an official trackback specification.

Pingbacks

Pingbacks were designed to solve some of the problems that people saw with trackbacks. The official pingback documentation makes pingbacks sound an awful lot like trackbacks:

> For example, Yvonne writes an interesting article on her Web log. Kathleen reads Yvonne's article and comments about it, linking back to Yvonne's original post. Using pingback, Kathleen's software can automatically notify Yvonne that her post has been linked to, and Yvonne's software can then include this information on her site.

There are three significant differences between pingbacks and trackbacks, though.

1. Pingbacks and trackbacks use drastically different communication technologies (XML-RPC and HTTP POST, respectively).
2. Pingbacks support auto-discovery where the software automatically finds out the links in a post, and automatically tries to pingback those URLs, while trackbacks must be done manually by entering the trackback URL that the trackback should be sent to.
3. Pingbacks do not send any content.

The best way to think about pingbacks is as **remote comments**:

- Person A posts something on his blog.
- Person B posts on her own blog, linking to Person A's post. This automatically sends a pingback to Person A when both have pingback enabled blogs.
- Person A's blog receives the pingback, then **automatically** goes to Person B's post to confirm that the pingback did, in fact, originate there.

The pingback is generally displayed on Person A's blog as simply a link to Person B's post. In this way, all editorial control over posts rests exclusively with the individual authors (unlike the trackback excerpt, which can be edited by the trackback recipient). The automatic verification process introduces a level of authenticity, making it harder to fake a pingback.

Some feel that trackbacks are superior because readers of Person A's blog can at least see some of what Person B has to say, and then decide if they want to read more (and therefore click over to Person B's blog). Others feel that pingbacks are superior because they create a verifiable connection between posts.

Verifying Pingbacks and Trackbacks

Comments on blogs are often criticized as lacking **authority**, since anyone can post anything using any name they like: there's no verification process to ensure that the person is who they claim to be. Trackbacks and Pingbacks both aim to provide some verification to blog commenting.

Comment Moderation

Comment Moderation is a feature which allows the website owner and author to monitor and control the comments on the different article posts, and can help in tackling comment spam. It lets you moderate comments, & you can delete unwanted comments, approve cool comments and make other decisions about the comments.

Comment Spam

Comment Spam refers to useless comments (or trackbacks, or pingbacks) to posts on a blog. These are often irrelevant to the context value of the post. They can contain one or more links to other websites or domains. Spammers use Comment Spam as a medium to get higher page rank for their domains in Google, so that they can sell those domains at a higher price sometime in future or to obtain a high ranking in search results for an existing website.

Spammers are relentless; because there can be substantial money involved, they work hard at their "job." They even build automated tools (robots) to rapidly submit their spam to the same or multiple weblogs. Many webloggers, especially beginners, sometimes feel overwhelmed by Comment Spam.

There are solutions, though, to avoiding Comment Spam. WordPress includes many tools for combating Comment Spam. With a little up front effort, Comment Spam can be manageable, and certainly no reason to give up weblogging.

Pretty Permalinks

Permalinks are the permanent URLs to your individual weblog posts, as well as categories and other lists of weblog postings. A permalink is what another weblogger will use to refer to your article (or section), or how you might send a link to your story in an e-mail message. Because others may link to your individual postings, the URL to that article shouldn't change. Permalinks are intended to be **permanent** (valid for a long time).

"Pretty" Permalinks is the idea that URLs are frequently visible to the people who click them, and should therefore be crafted in such a way that they make sense, and not be filled with incomprehensible parameters. The best Permalinks are "hackable," meaning a user might modify the link text in their browser to navigate to another section or listing of the weblog. For example, this is how the default Permalink to a story might look in a default WordPress installation:

```
/index.php?p=423
```

How is a user to know what "p" represents? Where did the number 423 come from?

In contrast, here is a well-structured, "Pretty" Permalink which could link to the same article, once the installation is configured to modify permalinks:

```
/archives/2003/05/23/my-cheese-sandwich/
```

One can easily guess that the Permalink includes the date of the posting, and the title, just by looking at the URL. One might also guess that hacking the URL to be `/archives/2003/05/` would get a list of all the postings from May of 2003. Pretty (cool). For more information on possible Permalink patterns in WordPress, see Using Permalinks.

Blog by email

Some blogging tools offer the ability to email your posts directly to your blog, all without direct interaction through the blogging tool interface. WordPress offers this cool feature. Using email, you can now send in your post content to a pre-determined email address & voila! Your post is published!

Post Slugs

If you're using Pretty Permalinks, the Post Slug is the title of your article post within the link. The blogging tool software may simplify or truncate your title into a more appropriate form for using as a link. A title such as "I'll Make A Wish" might be truncated to "ill-make-a-wish". In WordPress, you can change the Post Slug to something else, like "make-a-wish", which sounds better than a wish made when sick.

Excerpt

Excerpts are condensed summaries of your blog posts, with blogging tools being able to handle these in various ways. In WordPress, Excerpts can be specifically written to summarize the post, or generated automatically by using the first few paragraphs of a post or using the post up to a specific point, assigned by you.

Plugins

Plugins are cool bits of programming scripts that add additional functionality to your blog. These

are often features which either enhance already available features or add them to your site.

WordPress offers simple and easy ways of adding Plugins to your blog. From the Administraton Panel, there is a Plugin Page. Once you have uploaded a Plugin to your WordPress plugin directory, activate it from the Plugins Management SubPanel, and sit back and watch your Plugin work. Not all Plugins are so easily installed, but WordPress Plugin authors and developers make the process as easy as possible.

Basics-A Few Blogging Tips

Starting a new blog is difficult and this can put many people off, there are then other people who have blogs with no comments or visits. You want to stand out from this crowd of millions of bloggers, you want to be one of the few hundred thousand blogs that are actually visited. So here are some simple tips to help you on your way to blogging mastery:

1. Post regularly, but don't post if you have nothing worth posting about.
2. Stick with only a few specific genres to talk about.
3. Don't put 'subscribe' and 'vote me' links all over the front page until you have people that like your blog enough to ignore them (they're usually just in the way).
4. Use a clean and simple theme if at all possible.
5. Enjoy, blog for fun, comment on other peoples' blogs (as they normally visit back).

First Steps With WordPress

Now What?

You've just completed the famous 5 Minute Installation of WordPress without stress or fuss. WordPress is packed with many amazing features. So now that you've got it installed, what should you do?

Let's take a step-by-step tour through your WordPress site and learn about how all the different functions work and how to make your new site your own.

During the first part of this tutorial, we ask that *you don't change anything within the program*, unless it is part of the tutorial. Just follow these simple steps and soon you will be changing everything.

Starting from the Top

Begin by logging into the Administration Panel. This is the brain behind your website, the place where you can let your creativity explode, writing brilliant prose and designing the best and most lovely website possible. This is where the organization of your site begins - and this is just the start.

View Site Link

From the Administration Panel, from the top of the screen menu choose **View Site**. Like it? Don't like it? Doesn't matter, just look at it. This is where you are going to be spending a lot of time over the next few minutes, hours, weeks, months....

Test Driving Your WordPress Site

Take time to look at the site before you get into the changing of things and figuring out how all of this works; it's important to see how the default WordPress Theme is laid out and works. Consider this the test drive before you start adding on all the special features.

WordPress Default Theme - Kubrick

The layout you are looking at is called a Theme. It is the Presentation of your website, styling the look of the site. The default WordPress Theme features a blue "header" at the top with the title placeholder of your site. Along the side you will see some titles and links. This is your "sidebar menu." Within the main middle section of the page is the "post." At the bottom of the page is the "footer."

Let's look at the post for a moment. There is a title, and below the title is some information. This is called the *post meta data* and contains information about the post such as the date and time the post was made, the author, and the categories the post is in.

Scroll down the page and notice the bar at the end of the page. This is called the "footer," and for now it says "(your blog) is proudly powered by WordPress."

Back to the sidebar, you will see different sections with information. Among these you may find a list of Pages, Categories, Archives, Calendar, and Dates. This is part of the menu or navigation panel that people will use to move around your site, visiting posts from different categories or time periods.

It's All in the Details

Take time to notice the smaller details of this web page layout and design. Move your mouse over the title of the article post. Notice how it changes color. This is called a *hover*. Most Themes feature a distinctive color or change when you move your mouse over a link. Move your mouse over any of the links in the sidebar. Do they change? Is the change the same? You can change your link hovers to look different in different sections of your page, but typically they should be uniform. Also look at the color of the links. How are they colored to stand out from the rest of the text?

Observe the small design details and where they are placed within the page. In the near future, you may want to change some of these details, such as the color of the title in the blue box at the top of the page. If you remember that is called the *header* then you will know to look within the *header* section of your style sheet, the file that controls the look of your web page, when you want to make changes to it.

Take a Quick Trip Around

For now you only have one post. It is residing within a page that is laid out as your *home page* or the front page. If you click on the title of the post, it will take you to the specific page for that post. The first page or home page of your site features the most recent posts on your site. Each post title will link to the actual page of the post. Some Theme designers design their *single* post pages to look different from the home page. By clicking on the title, you are taken to another web page that looks different from the home page.

WordPress Default Theme - Single Post Look

Again, in the single post, pay attention to the layout and notice what is now different about the design elements. Is the header different? Smaller, larger, or a different color? Is there a sidebar? In the default Theme for WordPress, the sidebar disappears in the single post. Look at all the details and take note of the differences.

Posts are usually stored in **categories** so you can keep related topics together. Right now you only have one category, but will soon want more. Click on the single category that appears in the sidebar of the home page. You are now in a page that has been generated to display only the posts within that category. Again, take a look at the layout and see how it may be different from the home page and the single post.

Do the same with the **Archives**. You may only have one post, but look at how the pages are laid out. They may or may not change, but look at all of it to see how it all works.

All of these changes are created from only a few files called template files and you can learn more about how they work in Stepping Into Templates. For now, however, let's get on with how the rest of WordPress works.

Test Drive the WordPress Admin Panels

WordPress Admin Dashboard

Now that you have an idea of how your site looks and what the different layout sections are called, it's time to test drive the WordPress Administration. This is like familiarizing yourself with the dashboard of your new website. In fact, the first page you see after logging in is called The Dashboard.

The Dashboard is a new feature in WordPress v1.5. It helps to keep you up to date on new and interesting bits of information from the many WordPress resources. In the corner it also features a list of the most recent activity you've done on your site.

Across the top of the Admin screen is the main menu, which says:

- Dashboard
- Write
- Manage
- Links
- Presentation
- Plugins
- Users
- Options
- Logout (name)

User Profile Panel

The links in the above list will take you to a series of articles that will guide you step-by-step through every aspect of the Admin panels. You're anxious to get started, so for now, let's start with the Users panel.

Click on the **Users** tab. The screen will change and you will see the panel called Profile. This is where you will enter information about you, the author and administrator of the site. In the next tab called Authors and Users you can set up more authors. Let's stick with you for right now. Fill in the information and click **Update Profile** when done.

Now, let's look at the powerful feature functions of the WordPress Admin.

Quick Changing the Look

The Presentation Panel

The Presentation panel allows you to change the look of your site using Themes. Themes are presentation styles that completely change the look of your site. Designed by WordPress users, there are hundreds of themes available for you to choose from. In your Presentation panel, you will see two themes, classic and default. To try this quick-change process, simply select **Classic** and then click **View Site** to see how it looks. Wow, you have another look and nothing else on the site has changed. It's that easy.

Go back to the Presentation panel (**Back button** on your browser) and select **Default** to bring the design back to what you had. To see it again, click **View Site**, and there it is. Honestly, it is that simple.

Writing and Managing Posts

Back in the Admin panel, take a look at the Write panel, and the Manage panel. You can use the tabs under the Write Menu to write posts and Pages. Using the tabs under the Manage menu, you can manage the posts and Pages in your site.

Let's start by making your first test post in the Write Post tab.

Write Post Panel

If the screen looks a little intimidating, the Codex article on Writing a Post will take you step-by-step through the process of writing a post. Take a moment to read through the article and post

your first entry and then return to this article and we'll take you onto the next step.

If you are in a hurry, then simply fill in the blanks, one by one, in the post beginning with the title and then write a little test message in the post window. This is just for a test, so you can write anything you want. When you are done, click **PUBLISH** below the post entry window and it is done. You will then see a blank Write a Post screen and you're ready to write another one. Go ahead. But do only three to four entries. We have more exciting work ahead of us.

Now that you've gotten a feel for writing posts, you can view your posts by clicking View Site at the top of the screen. Now it's time to get down to the real work.

Planning Session

All good websites come from a good plan. Sounds redundant, but it's true. If you want to create a good and solid website, you need a good and solid plan. I know it's hard to do, and I know you want to keep poking and playing with this exciting program, but it is time to take a break away from your computer and turn to the old paper and pen. That's right, we're going back in time to when people actually wrote things down.

On a piece of notebook paper, or whatever is lying around, describe your site. Take five to twenty minutes to come up with a purpose for your site, or better yet, call it your Mission Statement.

Answer the following questions:

1. What am I going to do with this?
2. Who is going to read this?
3. What kinds of information will I be posting?
4. Why am I doing this?
5. Who am I doing this for?
6. How often am I going to be posting and adding information?

Now, compile this information into a paragraph so it looks like this:

This website will be dedicated to X, Y, and Z,

and cover the topics of A, B, and C. The audience will

be _____ _____ _____. I will be adding

posts every _____ about _____ _____ _____.

I am doing this because _____ _____ _____.

Using the Information

From this exercise, we've gathered a lot of information. We've uncovered information on how you

might layout and design your site. If you know your audience is mostly made up of young people under the age of 25, you will probably want a fashionable look ranging from wild colors and crazy graphics to dark foreboding tones. Something appropriate for that generation. If you are providing factual information about a serious subject, then you will probably want a more conservative look where the information is more important than a lot of pop and flash.

You probably already have a design idea in mind, or you will be copying over from your previous site, but take a moment to use this information to reconsider your design, and to see how what you want will work with the WordPress options.

You have also uncovered the possible categories for your site. The topics and subjects you will be covering are listed in your purpose statement. Let's say your purpose statement said,

> *"The website will be dedicated to providing news and information on computers, web pages, and the Internet and cover the topics of computer tips, web page design, and Internet news."*

Your topics are your categories. Write your categories down below your purpose paragraph and notes about your web page design.

Now, what subcategories might be under these topics? Under **Computer Tips**, you might want to segregate them by **Windows**, **Linux**, and **Mac**. Or maybe **Software** and **Hardware**. You can have sub-sub-categories, but let's stop with subcategories for right now. Write these down.

Remember the question about *why* you are doing this? Is it because you have valuable and timely information or knowledge to share, because you want to talk about a subject that interests you, or maybe because you just think it will be fun to do. Why not? Everyone's doing it!

Understanding the timeliness of the information you want to present on your site helps you organize the information on your website. Your website is organized by several different methods. If the date of *when* you posted the information is critical to the success of the page, then having links to your posts referenced by date is important. If the information itself is more important and timeless, then having your posts referenced by category is the best choice.

Have you noticed that you are starting to lay out your website? If you remember our earlier test drive of your new WordPress website, we examined the sidebar menu. This is the area where your past posts are organized. If you take another look (yes, you can go back to your computer for a moment), you will see the sidebar is laid out in a list by Archives by date, Categories by category, and may even feature a calendar (turned off in the Default Theme but visible in the Classic Theme).

As you lay out your website on paper, consider whether you want both categories and dates, or just one of them in your sidebar. What information you have and how you want to help the user find the information is critical to your website design.

What Information Do You Want to Share

As you think about what information the user will need to know, you have to consider what information you are willing to share with them. That information may include how to contact you, what the purpose of the site is, who you are, and what your expertise is.

WordPress v1.5 offers a new feature called Pages which makes the process of presenting this

information in an easier fashion. Pages, similar to posts, are most commonly used to present unchanging information such as Pages for **About Us**, **Contact Us**, **Sign Up for Our Mailing List**, and other *static* information. Before creating your individual Pages, you need to think about what information you would like the Page to hold. Write down the possible Page titles and describe the information you are willing to share online on each Page.

Comments

Part of the fun of WordPress is the ability to have viewers leave comments on your site. It creates a dynamic interchange between you and the viewer. Do you want comments on your posts? Comments on posts come in a variety of forums, from *pats on the back* (*Good job! Like the post!*) to extensive conversations and commentary about the posts turning into long chats. Or maybe you are seeking comments that add to the information you've posted. How you present your comment form, and whether you do or not, invites people to comment.

Responding to comments and moderating them can also take up a lot of time. If they are critical to your site, then include them and consider how you want them presented. Go back to your test site; the first post created at the time of installation includes a sample comment. You can even make a few comments yourself on the posts you created. Take a look at how they are laid out and consider how you might want them to look to fit into the design and layout of your site.

When you have reached your decision about how you want to handle comments, take time to read through the article on comments and WordPress discussion options to help you set those features.

With this basic information, you are ready to return to your computer and start laying out your site and setting it up.

Setting Your Site Up

Before you get to the graphic look of your site, let's do a little more administration to your site to set it up. Consider making your first plugin installation the Codex and Forum Searcher Plugin. It allows you to search both the WordPress Codex and WordPress Support Forum from your WordPress Administration Panels. Click on one of the search results and the page will open in a new window or tab so you can have the article or discussion open while working on WordPress. This will make your transition to WordPress a much gentler one with information right at your fingertips.

You can also work from this page by clicking on a link with a Right Click and opening the documents in a new window or tab, so you can read along as you work on your site.

Let's start with making those categories written down on your list.

Create Categories

In the **Manage > Categories** tab, click **Add Category** and fill in the information about your category. Continue to add your *parent* categories, going down the list. Hold off on entering sub-categories until all the main categories are entered.

NOTE: *You can add any new categories any time, but make a note of the fact that*

categories can be sorted in WordPress in two ways: by name (alphabetically) or by ID number. As you enter the categories, they are assigned an ID number. It is difficult to change this, so if you don't want your categories sorted alphabetically, enter them in the order you want to see them presented on the screen.

WordPress Admin Manage Categories

When you have the parent categories entered, enter your sub-categories. In the pull down menu for Parent Category, you can select the *parent* to the sub-category you are adding. When you view your categories in the Manage > Categories panel, you will see the categories listed like this:

Computer Tips

- **Windows**
- **Linux**
- **Mac**

Internet News
Web Page Design

- **Web Standards**
- **WordPress**

  - - **Plugins**
  - - **Themes**

Put Posts in Categories

Let's put some of your test posts into categories so you can see how this works.

WordPress Admin Manage Posts

From the **Manage > Category panel**, click on the tab for **Posts**. You should see the test posts you entered here. To the right are three links that say: **View - Edit - Delete**. Click on **Edit** to edit one of the posts. On the right side of the Edit Post screen you will now see your **Categories**. Choose one of them by clicking in the box next to it. Then scroll down the page and click SAVE. Repeat this for your other test posts, putting each one in a different category.

Now view your page by clicking **View Site** at the top of the **Admin** panel. Do you see the categories listed in the sidebar now? Great. If you are missing a category, that usually means that there are no posts in it. This is the default function of WordPress, so not to worry. When you add a post to the "missing" category, it will appear on your web pages. Click on one of the categories and you will be taken to a page for just that category. You should see the posts that went into that category. This is a generated *Category* page.

Now, click on the **Archives** for the month showing. Now you are visiting a generated page of your posts listed in chronological order for this month - well, specifically for today only. Two methods of finding the same information.

Preventing Spam

There is more to think about when it comes to having comments on your site. Unfortunately we live in a world where spam is a fact of life. It is recommended that you begin battling the comment spammers with the helpful article, Introduction to Dealing with Comment Spam.

What is Next

You've now done all the basics for your new WordPress website. You know how to write a post, create a category, and how to view your site's information by category and archive. You can start the customization process, and when you are done, don't forget to delete your test posts! Then start writing some wonderful information to share with your new-found public!

Customizing Your WordPress Site

Once you are familiar with how WordPress works, it's time to get creative and start customizing. The tutorial now splits into different subjects that require no order. From here on you can do whatever you want, adding and subtracting, perfecting and scrambling your site at will. The amount of effort you put into the site is now up to you. You can work with the two WordPress Themes that came with the installation, or seek out another Theme that better meets your needs. You can totally customize all the links and information, or get serious and completely re-design the entire site to do whatever you want. You have the basics, the rest is up to your imagination.

Finding a WordPress Theme
: Look for one that better suits the look you desire on your site.

Customizing the Look
: When you are ready to plunge into the code, you can customize the look and layout of the site through CSS and modifing the Themes (or create your own).

Enhance Your Site with Plugins

Plugins add function and sometimes fun to your site. There are hundreds of different plugins from adding custom links like related articles to your sidebar to adding weather reports.

WordPress Themes

There are hundreds of WordPress Themes to choose from. All do basically the same thing but graphically present the information in a myriad of ways. Choose a few that look interesting to you, and meet your audience's needs and your desires, and then test drive them following the test drive instructions above. Click through the whole site, the categories and archives as well as the individual posts to see how the Theme handles each one. The look may be nice on the front page, but if it handles things in a way you don't like on the single post, then you will have to dig into the code and make changes. Not ready for that, try another theme.

If you run into problems, check out the Codex's Troubleshooting Themes article.

Customizing The Look

If you are familiar with CSS, HTML, and even PHP and MySQL, consider customizing the Theme to your own needs. This is not for the timid, and it is for the informed and experienced. If you want to expand your web page design and development skills, WordPress can help:

- Using Themes
- Theme Development
- Stepping Into Templates
- Templates Files
- Blog Design and Layout
- CSS Overview, Tips, Techniques, and Resources
- FAQ - WordPress Layout
- Stepping Into Template Tags
- Template Tags
- CSS Troubleshooting
- CSS Fixing Browser Bugs

WordPress Plugins

Plugin Panel

WordPress Plugins are also known as add-ons or extensions. They are software scripts that add

functions and events to your website. They cover the gamut from up-to-date weather reports to simple organization of your posts and categories. Plugins are designed by volunteer contributors and enthusiasts who like challenges and problem solving. They are usually fairly simple to install through the WordPress Admin Plugin panel, just follow the instructions provided by the plugin author. Remember, these are free and non-essential. If you have any problems with plugins, contact the plugin author's website or plugin source first, then search the Internet for help with that specific plugin, and if you haven't found a solution, then visit the WordPress forums for more help.

- WordPress Plugin Repository
- WordPress Plugins
- Managing Plugins
- Plugins

Above and Beyond the Basics

The exciting thing about WordPress is that there are few limits. Thousands of people are using WordPress for blogging and for running their websites. All have a different look and different functions on their sites.

- Well-Designed Sites Using WordPress

What you do from here is up to you, but here are a few places to take that first step beyond the basics:

- WordPress Features
- Working with WordPress
- Using Pages
- Understanding the WordPress Loop
- Troubleshooting
- Using Permalinks
- Press It - Post to your site from the web instantly!

Getting More Help

- Codex Main Page
- FAQ - Frequently Asked Questions
- Getting More Help
- Using the Support Forums
- Troubleshooting

Introduction to WordPress Terminology

WordPress was created by the developers as **weblogging** or blogging software. A blog, as defined in the Codex Glossary, is an online journal, diary, or serial, published by a person or group of people. Many *blogs* are personal in nature, reflecting the opinions and interests of the owner. But, *blogs* are now important tools in the world of business, politics, and entertainment.

Blogs are a form of a Content Management System (CMS) which Wikipedia calls "a system used to organize and facilitate collaborative content creation." Both *blogs* and *Content Management Systems* can perform the role of a website (**site** for short). A *website* can be thought of as a collection of articles and information about a specific subject, service, or product, which may not be a personal reflection of the owner.

Terminology Related to Content

MySQL
Themes
Theme Development
Plugins

Administration
Administration Panels
Links
Link Categories
Registered Users
Roles and Capabilities
Comments
Comments Comments SubPanel
Comment Moderation
Discussion Settings
Spam
Combating Spam

Help
Finding WordPress Help
Troubleshooting
WordPress FAQ
Troubleshooting
WordPress Lessons
WordPress Support Forum
Help with Codex

The term **Word** in WordPress refers to the words used to compose posts. *Posts* are the principal element (or content) of a blog. The *posts* are the writings, compositions, discussions, discourses, musings, and, yes, the rantings of the blog's owner and guest authors. *Posts*, in most cases, are the reason a blog exists; without *posts*, there is no blog!

To facilitate the post writing process, WordPress provides a full featured authoring tool with modules that can be moved, via drag-and-drop, to fit the needs of all authors. The Dashboard QuickPress module makes it easy to quickly write and publish a post. There's no excuse for not writing.

Integral to a blog are the pictures, images, sounds, and movies, otherwise know as media. *Media* enhances, and gives life to a blog's content. WordPress provides an easy to use method of inserting *Media* directly into posts, and a method to upload Media that can be later attached to posts, and a Media Manager to manage those various *Media*.

An important part of the posting process is the act of assigning those posts to categories. Each post in WordPress is filed under one or more *categories*. *Categories* can be hierarchical in nature, where one category acts as a parent to several child, or grandchild, categories. Thoughtful *categorization* allows posts of similar content to be grouped, thereby aiding viewers in

the navigation, and use of a site. In addition to categories, terms or keywords called tags can be assigned to each post. *Tags* act as another navigation tool, but are not hierarchical in nature.

In turn, post categories and tags are two of the elements of what's called post meta data. *Post meta data* refers to the information associated with each post and includes the author's name and the date posted as well as the post categories. *Post meta data* also refers to Custom Fields where you assign specific words, or keys, that can describe posts. But, you can't mention *post meta data* without discussing the term **meta**.

Generally, meta means **"information about"**; in WordPress, *meta* usually refers to **administrative**-type information. So, besides *post meta data*, *Meta* is the HTML tag used to describe and define a web page to the outside world, like *meta tag keywords* for search engines. Also, many WordPress-based sites offer a *Meta* section, usually found in the sidebar, with links to login or register at that site. And, don't forget Meta Rules: The rules defining the general protocol to follow in using this Codex, or *Meta,* as in the MediaWiki namespace that refers to administrative functions within Codex. That's a lot of *Meta*!

After a post is made public, a blog's readers will respond, via comments, to that post, and in turn, authors will reply. *Comments* enable the communication process, that give-and-take, between author and reader. *Comments* are the life-blood of most blogs.

Finally, WordPress also offers a content management tool called a Page. *Pages* often present static information, such as "About Me", or "Contact Us", *Pages*. Typically "timeless" in nature, *Pages* should not be confused with the time-oriented objects called *posts*. Interestingly, a *Page* is allowed to be commented upon, but a *Page* cannot be categorized.

Terminology Related to Design

The *flexibility* of WordPress is apparent when discussing terminology related to the design of a WordPress blog. At the core of WordPress, developers created a programming structure named The Loop to handle the processing of posts. *The Loop* is the critical PHP program code used to display *posts*. Anyone wanting to enhance and **customize** WordPress will need to understand the mechanics of *The Loop*.

Along with The Loop, WordPress developers have created Template Tags which are a group of PHP functions that can be invoked by designers to perform an action or display specific information. It is the *Template Tags* that form the basis of the Template Files. *Templates (files)* contain the programming pieces, such as *Template Tags*, that control the structure and flow of a WordPress site. These files draw information from your WordPress MySQL database and generate the HTML code which is sent to the web browser. A Template Hierarchy, in essence the order of processing, dictates how *Templates* control almost all aspects of the output, including Headers, Sidebars, and Archives. *Archives* are a dynamically generated list of posts, and are typcially grouped by date, category, tag, or author.

Templates and Template Tags are two of the pieces used in the composition of a WordPress Theme. A *Theme* is the overall design of a site and encompasses color, graphics, and text. A *Theme* is sometimes called the **skin**. With the recent advances in WordPress, Theme Development has become a hot topic. WordPress-site owners have available a long list of *Themes* to choose from in deciding what to present to their sites' viewers. In fact, with the use of a Theme Switcher Revisited Plugin, WordPress designers can allow their visitors to select their own *Theme*.

Plugins are custom functions created to extend the core functionality of WordPress. The WordPress developers have maximized flexibility and minimized code bloat by allowing outside developers the opportunity to create their own useful add-on features. As evidenced by the Plugin Directory, there's a *Plugin* to enhance virtually every aspect of WordPress. A Plugin management tool makes it extremely easy to find and install Plugins.

Terminology for the Administrator

Another set of terms to examine are those involving the Administration of a WordPress site. A comprehensive set of Administration Panels enables users to **easily** administer and monitor their blog. A WordPress administrator has a number of powers which include requiring a visitor to register in order to participate in the blog, who can create new posts, whether comments can be left, and if files can be uploaded to the blog. An Administrator also defines Links and the associated Link Categories which are an important part of a blog's connection to the outside world.

Some of the main administrative responsibilities of a WordPress blog involve adding, deleting, and managing Registered Users. Administering users means controlling Roles and Capabilities, or permissions. **Roles** control what functions a registered *user* can perform as those functions can range from just being able to login at a blog to performing the role administrator.

Another chief concern for the blog administrator is Comment Moderation. Comments, also called discussions, are responses to posts left for the post author by the visitor and represent an important part of "the give and take" of a blog. But *Comments* must be patrolled for Spam and other malicious intentions. The WordPress Administration Comments SubPanel simplifies that process with easy-to-use screens which add, change, and delete Comments.

And not to be forgotten is the obligation for an administrator to keep their WordPress current to insure that the latest features, bugs, and security fixes are in effect. To accomodate administrators, WordPress has a simple Upgrade Tool to download and install the lastest version of WordPress. There's no excuse to not upgrade!

The Terminology of Help

The final set of *jargon* relates to **helping** you with WordPress. There are many *help* resources available to WordPress users; Getting More Help, Finding WordPress Help, Troubleshooting, and WordPress FAQ (frequently asked questions) are good starting points. Also Getting Started with WordPress will **jump-start** readers into the world of WordPress and the excellent WordPress Lessons provide in-depth **tutorials** on many of the aspects of using WordPress. Among the most important resources is the WordPress Support Forum where **knowledgeable** volunteers answer your questions and help solve any problems related to WordPress. And, of course, this Codex which is filled with hundreds of articles designed to make your WordPress experience a success!

History of the WordPress Name

Besides the technical terminology of WordPress, it's also interesting to know the history of the name, WordPress. The name "WordPress" was originally coined by Christine Selleck (see related post) in response to developer Matthew Mullenweg's desire to associate his new software project with printing presses. In this sense, **press** refers to the world of reporters,

journalists, columnists, and photographers. An aptly chosen name, because WordPress serves as the **printing press** that enables its users to *publish* their **words**. It's a good name, don't you think so?

More Information and Resources

See Also

- The WordPress Glossary
- WordPress Features
- Advanced Topics
- Mailing Lists
- WordPress Support Forum
- Popular Codex Articles
- All Codex Articles

External Links

- World Wide Web @ Wikipedia
- Tim Berners-Lee creator of the World Wide Web @ Wikipedia
- The Semantic Web @ Wikipedia
- World Wide Web Consortium (W3C) homepage
- About the World Wide Web Consortium
- Web Standards Project

New To WordPress - Where to Start

If you are new to WordPress and you're worried about where to start, you've come to the right place! Here is a very simple step-by-step plan for getting started with WordPress. Please remember, if you need help along the way, plenty of options for assistance are listed in this article. Welcome to the exciting world of WordPress!

Step One - Read

Before you invest your valuable time and energy into installing WordPress, there are some documents you need to read. WordPress is a great product; it's easy-to-use, it's quite powerful, but it isn't necessarily the right software for everyone. Just like building a house, you have to use the right tool for the right job. Consider creating a PDF to read at your leisure.

- About Weblogs - What is Blogging all about?
- What is WordPress?
- WordPress Features
- Before You Install WordPress

Step Two - Make a Plan

Based upon the information you've just read, including instructions on installing WordPress, you should have a list of the things you need, and the things you need to do. If not, make that list now--you'll want to make sure it includes the following information:

- Website Host Requirements Checked and Verified
- Versions of PHP and MySQL Checked and Verified
- Web Host Compatibility with New Versions of WordPress
- Your Website Username and Password
- Text Editor Software
- An FTP Client Software
- Your Web Browser of Choice

The following documents will help you understand more about how WordPress works and how to make a plan for your WordPress site:

- WordPress Features
- First Steps With WordPress
- WordPress Lessons

It is important to make a plan about how you want to use WordPress on your site. Here are some questions to ask yourself. Make a list of the answers so you can add to your plan.

- Will you install WordPress in the root directory, subdirectory, or you just want to make a test site to make sure you want to use it?
- Have you made a list of your site Categories? Understand that WordPress can only order Categories alphabetically by name or by ID (order entered through the Manage> Categories screen), so if the display order of your Categories is important to you, start

making your list of Categories.
- Have you made a list of Pages you may want to add to your site, such as **About**, **Contact**, or **Events**?

Step Three - Install WordPress

With this information and your plan, it's time to install WordPress.

- Before You Install WordPress
- Installing WordPress
- Hosting WordPress
- Editing the `wp-config.php` file
- Frequently Asked Questions About Installing WordPress
- Using FTP Clients and Software
- Changing File Permissions
- Upgrading WordPress
- Common Installation Problems
- Trouble: I Can't Login

Step Four - Set Up WordPress

With your installation complete, it's time to set up WordPress so it will work the way you want it to work. As you change various settings, it is recommended you view how those changes impact your site by frequently clicking the `View Site` link at the top of the Administration Screen. Though you may choose to do these steps in any order, your site will cause you fewer problems if you proceed in the following order:

- Administering Your Blog
- Users> Your User Profile - set the user information you want published on your site
- Your User Profile> Other Users - add authors and users that will be using your site, if applicable
- Options> General - set your site name and other site information
- Options> Writing - set the settings of your Write Post screen
- Options> Reading - set how many posts to show on the front page and in categories and your feed requirements
- Options> Discussion - Turn on or off comments and set how to handle them
- Manage> Categories - add a few categories to get started from your category list
- Manage> Posts - After you have written a few posts, this is where you will manage them by editing or deleting
- Presentation> Themes - maybe change the look of your site?
- Manage> Pages - add a Page or two like "About Us" or "Contact Me"
- Write> Write Post - start adding content to your site
- Writing Posts - step-by-step instructions on writing posts

Take time to explore the WordPress Codex site, the official documentation site for WordPress. You'll find helpful information by reading WordPress Lessons, and these helpful documents:

- Introduction to Dealing with Comment Spam
- Moderating Comments
- Using the Links Manager

- WordPress in Languages Other than English

Presentation and Themes

With the new WordPress version 1.5, changing the look of your WordPress website is possible with just a few clicks. Here is a list of resources and information about changing the look of your site with WordPress Themes.

- Using WordPress Themes
- Blog Design and Layout
- Using Pages

At this point, there may be something about your Theme choice that is bothering you, or, you really want to get your hands dirty understanding how your WordPress Theme works. These simple guides to help customize your WordPress Theme:

- Lessons: Designing Your WordPress Site
- CSS Overview, Tips, Techniques, and Resources
- Stepping Into Templates
- Lessons: Template Files
- Stepping Into Template Tags
- Lessons: Working With Template Tags
- WordPress Template Tags
- Understanding the WordPress Loop
- The WordPress Loop in Action
- Editing Files in WordPress
- Frequently Requested Design Help
- Frequently Asked Questions about Site Layout and Design

If you want to create a new WordPress Theme from scratch, or do major renovations, or even design WordPress Themes for public release, you will need to be familiar with HTML, XHMTL, and CSS. The following documents will get you started:

- Developing Your Own WordPress Theme
- Designing Themes for Public Release
- Validating a Website
- Lessons: Website Development
- CSS Fixing Browser Bugs
- CSS Troubleshooting
- Positioniseverything
- Position is Everything's 3 Complex Column - Perched on a Lily Pad
- Position is Everything Piefecta 3-Column Layout

If you want a custom-made WordPress Theme created especially for you by expert web-designers, it is recommended you search for qualified web-designers on the Internet, or look in your local community, or draw from the List of Recommended Web Page Designers by Laughing Squid.

Adding Plugins

There are many "add-on" scripts and programs for WordPress called Plugins that add more

capabilities, choices, and options to your WordPress site. WordPress Plugins do many things, including; customizing the results of your site information, adding weather reports, adding spell check capability, and presenting custom lists of posts and acronyms. For more on how to work with Plugins and where to find WordPress Plugins for your site:

- Managing Plugins
- Plugins
- http://www.wp-plugins.org/
- http://www.wp-plugins.net/
- Bloggingpro List of Plugins

Advanced Use of WordPress

Now that you are familiar with the basic features and functions of how WordPress works, it might be time for you to plunge deeper into the power of WordPress. The links below will expand your familiarity with PHP, HTML, XHTML, and CSS:

- Lessons: WordPress Feature and Functions
- Lessons: WordPress Tech Techniques
- Using Permalinks
- Photoblogs and Galleries
- WordPress Advanced Techniques
- Advanced Techniques for Plugins and Customization
- WordPress Server and Database Information
- Developer Documentation

Need More Help

As simple and easy as it is to use WordPress, if troubles arise, if something is confusing, if things aren't working, don't despair because help is available! Even though WordPress is free and open source, there are literally hundreds of volunteers eager to help you. Here are some helpful resources for WordPress:

- FAQ
- Getting More Help
- Using the Support Forums
- WordPress Forum
- IRC Freenode WordPress Support on channel #wordpress
- WordPress IRC Live Help

And Finally

Now that you're a full fledged WordPress user, consider contributing to the WordPress Codex, Support Forum, Development, and other volunteer efforts that keep WordPress going. WordPress is free and totally supported by volunteers, and your help is needed.

Using Themes

Before the advent of Themes, WordPress generated content using a single file, `index.php`, and files to support comment display and submission. A single style sheet controlled the presentation. All other pages, including the category and archive pages, were generated by passing parameters to the `index.php` page.

The new Theme system provides two convenient features.

Physically Separate Components
: The new WordPress modular template files system provides a method to define separate physical PHP files for the different components of your WordPress site. This allows creation of unique designs and functionality for many special pages, such as category archives, monthly archives, and the individual entry pages.

Quickly Change Layout and Design
: It allows users with appropriate permissions to quickly change the layout of the entire site by uploading a new theme and essentially flipping a switch in the admin panel.

The old method of generating pages will still work. If you are upgrading from v1.2 or v1.2.1 or v1.2.2, **you can continue using your existing design**. Keeping your older design will not prevent you from adding additional themes and easily switching between designs.

What is a Theme?

Fundamentally, the WordPress Theme system is a way to "skin" your weblog. Yet, it is more than just a "skin." Skinning your site implies that only the design is changed. WordPress Themes can provide much more control over the look *and presentation* of the material on your website.

A WordPress Theme is a collection of files that work together to produce a graphical interface with an underlying unifying design for a weblog. These files are called **template files**. A theme modifies the way the site is displayed, without modifying the underlying software. Themes may include customized template files, image files (`*.jpg`, `*.gif`), style sheets (`*.css`), custom Pages, as well as any necessary code files (`*.php`). For an introduction to template files, see Stepping Into Templates.

Themes are a whole new ball game. Let's say you write a lot about cheese and gadgets. Through the innovative use of the WordPress Loop and template files, you can customize your Cheese category posts to look different from your Gadgets category posts. With this powerful control over what different pages and categories look like on your site, you are limited only by your imagination. For information on how to use different Themes for different categories or posts, see The Loop in Action and Category Templates.

Get New Themes

The WordPress Theme Directory is the official site for WordPress Themes which have been checked and inspected, and are free for downloading. The site features the ability to search by type and style, and offers a demonstration of the page view elements of the Theme.

Using Themes

WordPress supplies two themes in its distribution for your initial use. You can switch between these two themes using the admin panel. Themes that you add to that directory will appear in the Administration Panels> Design> Themes as additional selections.

Adding New Themes

There are many themes available for download that will work with your WordPress installation.

Presentation Theme Panel
If the theme that you are installing provides instructions, be sure to read through and follow those instructions for the successful installation of the theme. **It is recommended that theme developers provide installation instructions for their own themes**, because themes can provide special optional functionality that may require more steps than the basic installation steps covered here. If your theme does not work after following any provided instructions, please **contact the theme author for help**.

To add a new theme to your WordPress installation, follow these basic steps:

1. Download the theme archive and extract the files it contains. You may need to preserve the directory structure in the archive when extracting these files. Follow the guidelines provided by your theme author.
2. Using an FTP client to access your host web server, create a directory to contain your theme in the `wp-content/themes` directory provided by WordPress. For example, a theme named **Test** should be in `wp-content/themes/test`. Your theme may provide this directory as part of the archive.
3. Upload the theme files to the new directory on your host server.
4. Follow the instructions below for selecting the new theme.

Adding New Themes in cPanel

If your host offers the cPanel control panel, and the theme files are in a .zip or .gz archive follow these instructions. Note: This assumes the theme you download is a compressed (zip) file and the files in the zip file are in their 'named' folder.

1. Download the theme zip file to your local machine.
2. In cPanel File Manager, navigate to your themes folder. If you have WordPress installed in it's own folder called wordpress, you would navigate to "public_html/wordpress/wp-content/themes" and if WordPress is installed in your web-root folder you would navigate to "public_html/wp-content/themes".
3. Once you've navigated to the themes folder in cPanel File Manager, click on Upload file(s) and upload that zip file you saved in Step 1.

4. Once the zip file is uploaded, click on the zip file name in cPanel, then in the panel to the right, click on Extract File Contents, and that zip file will be uncompressed.
5. Follow the instructions below for selecting the new theme.

Note: You can also install and activate the plugin Get_Theme, to download themes directly to your blog.

Selecting the Active Theme

To select the active theme for your site:

1. Log in to the WordPress Administration Panels.
2. Select the Design subpanel, then Themes.
3. From the **Available Themes** section, click on theme title (or theme screenshot) for the theme you wish to activate.
4. A preview of the theme will be shown to activate theme click the **Activate "Theme Name"** link in the top right.

Your selection should immediately become active.

Creating Themes

If you are interested in creating your own theme for distribution, or learning more about the architecture of themes, please review the documentation regarding Theme Development and Designing Themes for Public_Release.

Theme Files

The following are the files typically included within a Theme.

- 404 Template = `404.php`
- Archive Template = `archive.php`
- Archive Index Page = `archives.php`
- Comments Template = `comments.php`
- Footer Template = `footer.php`
- Header Template = `header.php`
- Links = `links.php`
- Main Template = `index.php`
- Page Template = `page.php`
- Popup Comments Template = `comments-popup.php`
- Post Template = `single.php`
- Search Form = `searchform.php`
- Search Template = `search.php`
- Sidebar Template = `sidebar.php`
- Stylesheet = `style.css`

== Moving from 1.2.x to 1.5 ==

For details on upgrading WordPress and your WordPress Theme from v1.2 to 1.5, see: Upgrade_1.2_to_1.5.

Theme Tools and Other Resources

These tools and resources will aid you in creating and/or enhancing themes.

Layout

- Blog Design and Layout
- Stepping Into Templates
- Customizing Your Sidebar
- Good Navigation Links
- Next and Previous Links
- Styling Lists with CSS
- Creating Horizontal Menus
- Dynamic Menu Highlighting
- FAQ Layout and Design
- WordPress Index Builder
- Photoshop Template for the WordPress Default Theme

Templates and Template Tags

- Template Files
- Stepping Into Templates
- Stepping Into Template Tags
- Template Tags
- Developing WordPress Themes
- The Loop in Action
- Template Tag Layout in simple to read form
- Anatomy of a WordPress Theme
- Templates and the is functions
- Dissection of a WordPress Theme

Tools and Resources

- Designing Themes for Public Release
- Know Your Sources
- Validating a Website
- Finding Your CSS Styles
- CSS Fixing Browser Bugs
- How to convert XHTML/HTML/CSS to Wordpress
- WordPress Design Sandbox Article
- Working On CSS and the WordPress Theme (using a sandbox)
- From XHTML CSS to WordPress (making a Theme}
- Tutorial: How to Create a Wordpress Theme
- Coevolving Innovations guide to Installing Wordpress (including a Theme) on a web host with Fantastico and cPanel (at Wordpress 2.2.1)
- Tutorial: How to Install a Wordpress Theme
- Working with Multiple Themes Outside of the WordPress Installation Directory
- Theme Test Drive plugin to preview themes
- Generate a custom theme for beginners

Colors, Graphics, and Fonts

- Kubrickr - Changes Default Theme Header Images
- Header Graphics
- Creative Commons Images
- Image * After - Free Stock Image Library
- Free Icons Library
- Stock.xchng - Free Stock Image Library
- Playing With Fonts
- Developing a Colour Scheme
- Colr.org - For Finding Colors in an Image
- I Like Your Colors
- List of More than 70 Free Stock Photo Sites
- kuler - Create color palettes with this web app from Adobe Labs

Theme Development

The following article is about developing or designing your own WordPress Theme. If you wish to learn more about how to install and use Themes, review the documentation regarding Using Themes. This topic differs from Using Themes because it discusses the technical aspects of writing code to build your own Themes rather than how to activate Themes or where to obtain new Themes.

You may wish to develop WordPress Themes for your own use or for distribution.

Why WordPress Themes

WordPress Themes are files and styles that work together to create a presentation or look for a WordPress site. Each Theme may be different, offering many choices for users to take advantage of in order to instantly change their website look. Why should you build your own WordPress Theme?

- To create your own unique WordPress site look
- To take advantage of templates, template tags, and the WordPress Loop to generate different web page results and looks.
- To provide alternative templates for specific site features, such as category pages and search result pages.
- To quickly switch between two site layouts, or to take advantage of a Theme or style switcher to allow users to change the look of your site.
- To design WordPress Theme(s) so that others may enjoy your designs through public release.

A WordPress Theme has many benefits, too.

- It separates the presentation styles and template files from the system files so the site will upgrade without drastic changes to the visual presentation of the site.
- It allows for customization of the presentation and web page results unique to that Theme.
- It allows for quick changes of the look and feel of a WordPress site.
- It takes away the need for a WordPress user to have to learn CSS, HTML, and PHP in order to have a good looking website.

Why should you build your own WordPress Theme? That's the real question.

- It's an opportunity to learn more about CSS, HTML/XHTML, and PHP.
- It's an opportunity to put your expertise with CSS, HTML/XHTML, and PHP to work.
- It's creative.
- It's fun (most of the time).
- If you release it to the public, you can feel good that you shared and gave something back to the WordPress Community (okay, bragging rights!)

Anatomy of a Theme

WordPress Themes live in subdirectories residing in `wp-content/themes/`. The Theme's subdirectory holds all of the Theme's style sheet files, template files, and optional functions file (`functions.php`), and images. For example, a Theme named "test" would probably reside in the directory `wp-content/themes/test/`.

WordPress includes two Themes in the download, a "Classic" and "Default" Theme. The two Themes are different and use different functions and tags to generate their web page results and looks. Examine the files carefully for these Themes to get a better idea of how to build your own Theme files.

WordPress Themes consist of three main types of files, in addition to images. One is the style sheet called `style.css`, which controls the presentation (look) of the web pages. The second is the optional functions file (`functions.php`). The other files are the template files which control the way the web page generates the information from the Database to be displayed as a web page. Let's look at these individually.

Theme Style Sheet

In addition to CSS style information for your theme, the stylesheet, `style.css` ***must*** provide details about the Theme in the form of comments. ***No two Themes are allowed to have the same details*** listed in their comment headers, as this will lead to problems in the Theme selection dialog. If you make your own Theme by copying an existing one, make sure you change this information first.

The following is an example of the first few lines of the stylesheet, called the style sheet header, for the Theme "Rose":

```
/*   Theme Name: RoseTheme URI: the-theme's-homepageDescription: a-brie
f-descriptionAuthor: your-nameAuthor URI: your-URITemplate: use-this-to
-define-a-parent-theme--optionalVersion: a-number--optional.General com
ments/License Statement if any..*/
```

The simplest Theme includes only a `style.css` file, plus images, if any. To create such a Theme, you must specify a set of templates to *inherit* for use with the Theme by editing the `Template:` line in the `style.css` header comments. For example, if you wanted the Theme "Rose" to inherit the templates from another Theme called "test", you would include `Template: test` in the comments at the beginning of Rose's `style.css`. Now "test" is the parent Theme for "Rose", which still consists only of a `style.css` file and the concomitant images, all located in the directory `wp-content/themes/Rose`. Additionally (as of WordPress 2.7), the child theme may contain template files, which can be selected in the admin panel as normal, and will override the parent's template files where those possess the same name.

The comment header lines in `style.css` are required for WordPress to be able to identify a Theme and display it in the Administration Panel under Design> Themes as an available Theme option along with any other installed Themes.

Note : *When defining the parent Theme, in the `Template:` section of the comment header, you must use the name of the directory of the style. For example, to use as parent template the Default Wordpress Theme, don't write `Template: WordPress Default`, but `Template: default`, because default is the directory of this Theme.*

Theme Functions File

A theme can optionally use a functions file, which resides in the theme subdirectory and is named `functions.php`. This file basically acts like a plugin, and if it is present in the theme you are using, it is automatically loaded during WordPress initialization (both for admin pages and external pages). Suggested uses for this file:

- Define functions used in several template files of your theme
- Set up an admin screen, giving users options for colors, styles, and other aspects of your theme

The "Default" WordPress theme contains a `functions.php` file that defines functions and an admin screen, so you might want to use it as a model. Since `functions.php` basically functions as a plugin, the Function Reference list is the best place to go for more information on what you can do with this file.

Theme Template Files

Templates are PHP source files used to generate the pages requested by visitors. Let's look at the various templates that can be defined as part of a Theme.

WordPress allows you to define separate templates for the various aspects of your weblog; however, it is not essential to have all these different template files for your blog to function fully. Templates are chosen and generated based upon the Template Hierarchy, depending upon what templates are available in a particular Theme. As a Theme developer, you can choose the amount of customization you want to implement using templates. For example, as an extreme case, you can use only one template file, called `index.php` as the template for *all* pages generated and displayed by the weblog. A more common use is to have different template files generate different results, to allow maximum customization.

Basic Templates

At the very minimum, a WordPress Theme consists of two files:

- `style.css`
- `index.php`

Both of these files go into the Theme's directory. The `index.php` template file is very flexible. It can be used to include all references to the header, sidebar, footer, content, categories, archives, search, error, and other web pages generated by the user on your site. Or it can be *subdivided* into modular template files, each one taking on part of the workload.If you do not provide any other template files, WordPress will use the built-in default files. For example, if you do not have either a `comments.php` or `comments-popup.php` template file, then WordPress will automatically use the `wp-comments.php` and `wp-comments-popup.php` template files using Template Hierarchy. These default templates may not match your Theme very well, so you probably will want to provide your own. The basic files normally used to subdivide (which go into the Theme's directory) are:

- `header.php`
- `sidebar.php`
- `footer.php`

- `comments.php`
- `comments-popup.php`

Using these modular template files, you can put template tags within the `index.php` master file to include or *get* these units where you want them to appear in the final generated web page.

- To include the header, use the `get_header()` template tag.
- To include the sidebar, use the `get_sidebar()` template tag.
- To include the footer, use the `get_footer()` template tag.

Here is an example of the *include* usage:

```
<?php get_sidebar(); ?><?php get_footer(); ?>
```

For more on how these various Templates work and how to generate different information within them, read the Templates documentation.

Query-based Templates

WordPress can load different Templates for different *query* types. There are two ways to do this: as part of the built-in Template Hierarchy, and through the use of Conditional Tags within The Loop of a template file.

To use the Template Hierarchy, you basically need to provide special-purpose Template files, which will automatically be used to override `index.php`. For instance, if your Theme provides a template called `category.php` and a category is being queried, `category.php` will be loaded instead of `index.php`. If `category.php` is not present, `index.php` is used as usual.

You can get even more specific in the Template Hierarchy by providing a file called, for instance, `category-6.php` -- this file will be used rather than `category.php` when generating the page for the category whose ID number is 6. (You can find category ID numbers in Manage> Categories if you are logged in as the site administrator in WordPress version 2.3 and below. In WordPress 2.5 the ID column was removed from the Admin panels. You can locate the category id by clicking 'Edit Category' and looking on the URL address bar for the cat_ID value. It will look '...categories.php?action=edit&cat_ID=3' where '3' is the category id). For a more detailed look at how this process works, see Category Templates.

If your Theme needs to have even more control over which Template files are used than what is provided in the Template Hierarchy, you can use Conditional Tags. The Conditional Tag basically checks to see if some particular condition is true, within the WordPress Loop, and then you can load a particular template, or put some particular text on the screen, based on that condition.

For example, to generate a distinctive style sheet in a post only found within a specific category, the code might look like this:

```
<?phpif (is_category(9)) {   // looking for category 9 posts   include(
TEMPLATEPATH . '/single2.php');} else {   // put this on every other ca
tegory post   include(TEMPLATEPATH . '/single1.php');}?>
```

Or, using a query, it might look like this:

```
<?php$post = $wp_query->post;if ( in_category('9') ) {   include(TEMPLA
TEPATH . '/single2.php');} else {   include(TEMPLATEPATH . '/single1.ph
p');}?>
```

In either case, this example code will cause different templates to be used depending on the category of the particular post being displayed. Query conditions are not limited to categories, however -- see the Conditional Tags article to look at all the options.

Media Icons

This feature is currently broken in WordPress 2.5.

Wordpress uses media icons to represent attachment files on your blog and in the Admin interface, if those icons are available.

It looks for image files named by media type in the `images` directory of the current theme. (As of Wordpress 2.2, the default theme comes with only one media icon, `audio.jpg`.)

For example, for an attachment of MIME type `audio/mpeg`, Wordpress would look for an icon file at these locations, stopping after the first match (see `wp_mime_type_icon`):

1. *my_theme*/images/audio.jpg
2. *my_theme*/images/audio.gif
3. *my_theme*/images/audio.png
4. *my_theme*/images/mpeg.jpg
5. *my_theme*/images/mpeg.gif
6. *my_theme*/images/mpeg.png
7. *my_theme*/images/audio_mpeg.jpg
8. *my_theme*/images/audio_mpeg.gif
9. *my_theme*/images/audio_mpeg.png

Theme Template Files List

Here is the list of Theme template files recognized by WordPress. Of course, your Theme can contain any other style sheets, images, or files. *Just keep in mind that the following have special meaning to WordPress -- see Template Hierarchy for more information.*

`style.css`
: The main stylesheet. This **must** be included with your Theme, and it must contain the information header for your Theme.

`index.php`
: The main template. If your Theme provides its own templates, `index.php` must be present.

`comments.php`
: The comments template. If not present, `comments.php` from the "default" Theme is used.

`comments-popup.php`
: The popup comments template. If not present, `comments-popup.php` from the "default" Theme is used.

`home.php`
: The home page template.

`single.php`
: The single post template. Used when a single post is queried. For this and all other query templates, `index.php` is used if the query template is not present.

`page.php`
: The page template. Used when an individual Page is queried.

`category.php`
: The category template. Used when a category is queried.

`author.php`
: The author template. Used when an author is queried.

`date.php`
: The date/time template. Used when a date or time is queried. Year, month, day, hour, minute, second.

`archive.php`
: The archive template. Used when a category, author, or date is queried. Note that this template will be overridden by `category.php`, `author.php`, and `date.php` for their respective query types.

`search.php`
: The search results template. Used when a search is performed.

`404.php`
: The **404 Not Found** template. Used when WordPress cannot find a post or page that matches the query.

These files have a special meaning with regard to WordPress because they are used as a replacement for `index.php`, when available, according to the Template Hierarchy, and when the corresponding Conditional Tag (a.k.a `is_*();` function) returns true. For example, if only a single post is being displayed, the `is_single()` function returns 'true', and, if there is a `single.php` file in the active Theme, that template is used to generate the page.

Referencing Files From a Template

The WordPress Default Theme (based on Michael Heilemann's Kubrick layout for WordPress 1.2) provides a good example of how queries are mapped onto templates.

The code `<?php bloginfo('template_directory'); ?>` inserts the URL of the template directory into the template output. You can append any additional URI information to this output to reference files in your Theme.

The code `<?php bloginfo('stylesheet_directory'); ?>` inserts the URL of the directory that contains the current Theme stylesheet into the template output. You can append any additional URI information to this output to reference files for your Theme, specifically those that are used by the stylesheet. (this has been deprecated, use `<?php bloginfo('stylesheet_directory'); ?>` instead.)

The constant TEMPLATEPATH is a reference to the absolute path to the template directory for the current Theme (without the / at the end).

Note that URIs that are used in the stylesheet are relative to the stylesheet, not the page that references the stylesheet. This obviates the need to include PHP code in the CSS file to specify directories. For example, if you include an `images/` directory in your Theme, you need only specify this relative directory in the CSS, like so:

```
h1 { background-image: URL(images/my_background.jpg); }
```

It is a good practice to use URIs in the manner described above to reference files from within a template, since, then your template will not depend on absolute paths.

Defining Custom Templates

It is possible to use the WordPress plugin system to define additional templates that are shown based on your own custom criteria. This advanced feature can be accomplished using the `template_redirect` action hook. More information about creating plugins can be found in the Plugin API reference.

Plugin API Hooks

When developing Themes, it's good to keep in mind that your Theme should be set up so that it can work well with any WordPress plugins you (or another Theme user) might decide to install. Plugins add functionality to WordPress via "Action Hooks" (see Plugin API for more information). Most Action Hooks are within the core PHP code of WordPress, so your Theme does not have to have any special tags for them to work. But a few Action Hooks do need to be present in your Theme, in order for Plugins to display information directly in your header, footer, sidebar, or in the page body. Here is a list of the special Action Hook Template Tags you need to include:

wp_head
: Goes in the HTML `<head>` element of a theme; `header.php` template. Example plugin use: add javascript code.

  Usage: `<?php do_action('wp_head'); ?>`
  -or- `<?php wp_head(); ?>`

wp_footer
: Goes in the "footer" of a theme; `footer.php` template. Example plugin use: insert PHP code that needs to run after everything else, at the bottom of the footer.

  Usage: `<?php do_action('wp_footer'); ?>`
  -or- `<?php wp_footer(); ?>`

wp_meta
: Typically goes in the `<li>Meta</li>` section of a theme's menu or sidebar; `sidebar.php` template. Example plugin use: include a rotating advertisement or a tag cloud.

  Usage: `<?php do_action('wp_meta'); ?>`
  -or- `<?php wp_meta(); ?>`

comment_form
: Goes in `comments.php` and `comments-popup.php`, directly before the comment form's closing tag (`</form>`). Example plugin use: display a comment preview.

  Usage: `<?php do_action('comment_form', $post->ID); ?>`

For a real world usage example, you'll find these plugin hooks included in the default theme's templates.

Theme Development General Guidelines

Please be clear about the following in your documentation (a README file included with your Theme helps many users over any potential stumbling blocks):

1. Indicate precisely what your Theme and template files will achieve.
2. Adhere to the naming conventions of the standard theme hierarchy.
3. Indicate deficiencies in your Themes, if any.
4. Clearly reference any special modifications in comments within the template and style sheet files. Add comments to modifications, template sections, and CSS styles, especially those which cross template files.
5. If you have any special requirements, which may include custom Rewrite Rules, or the use of some additional, special templates, images or files, please explicitly state the steps of action a user should take to get your Theme working.
6. Try and test your Theme across browsers to catch at least a few of the problems the users of the Theme may find later.
7. Provide contact information (web page or email), if possible, for support information and questions.

Take time to read through Designing Themes for Public Release, an article with good tips on preparing your Theme for the public.

References and Resources

There is a comprehensive list of WordPress Theme and Template File resources in the Templates article.

FAQ Layout and Design

Layout and Styles

I am having trouble with my CSS so where can I find help?

The following are articles that will help you troubleshoot and solve many of your CSS problems:

- Blog Design and Layout
- Finding Your CSS Styles
- CSS Fixing Browser Bugs
- CSS Troubleshooting
- WordPress CSS Information and Resources

How can I choose different styles or colours for my comments?

There are a variety of WordPress Plugins that change the look, layout, and colors of your comments and comment form. Look for various Comments Plugins in the Official WordPress Plugin Directory.

How do I change the size of the popup comments window?

To change the look of the Popup Comments window in WordPress version 1.5, make changes to the `comment-functions.php` file where it shows the following line: `function comments_popup_script($width=400, $height=400, $file=) {.`

To change the look of the Popup Comments window in WordPress version 1.2.1 Mingus, make the following change to the `template-functions-comment.php` on line 50:

```
function comments_popup_script($width=400, $height=400, $file='wp-comme
nts-popup.php')
```

You can also change Line 81 of *wp-comments-popup.php* to alter the textarea size for people entering comments.

Where can I find some other Themes and templates to use for styling my blog?

- WordPress Theme Viewer
- Blogging Pro features a whole category of WordPress Themes
- The "official" WordPress Theme List
- Alex King's WordPress Theme Competition

Where can I find information about styling lists and nested lists?

See Styling Lists with CSS

How do I change the way menu links are listed?

See Styling Lists with CSS

How do I get rid of the bullet points next to my links?

See Styling Lists with CSS

How can I create horizontal menus?

See Creating Horizontal Menus

How can I get my categories to display in the order I want?

Two tags are involved in generating your category list. For more information on setting the order and look of these see list_cats() and wp_list_cats().

How can I get my links to open in a new window?

Opening links in a new window is considered bad form in today's web as it has been abused. Yet, it still serves a purpose for demonstration sites that require more than one window open at a time. This method will work for those links that you enter into the body of a post.

After entering the link using the Quicktags button for "link", add `target="_blank"` to the individual*- link you want to have open in a new window when clicked. Consider adding text indicating that this link will open a new window, as required by web accessibility standards.

```
<a href="http://example.com/page.php" title="Page Title - opens in new window" target="_blank">Page Title (Opens in new window)</a>
```

Is there a tool to encode *HTML* entities and tags so I can display code on my weblog?

The article Writing Code in Your Posts will help you write programming code and code examples in your posts. The Encode tool will convert your HTML/XHTML code into a form that can be displayed on your blog without it being treated as HTML by browsers.

There are also WordPress Plugins and other tools available to help integrate this proces into your site if you use it frequently to display code.

- Scott Yang's Syntax Highlighting with Enscript in WordPress
- Owen Winkler's Code Filter

Also see: Fun Character Entities

How do I do a *dropcaps* on the first letter of a post?

DropCaps is the name for the effect where the first letter of the first paragraph in an article drops below the line of text, and is displayed in a larger font-size than the other normal letters.

This can be done using BBCode quicktags. First, add this to your *style sheet*:

```
 #fp:first-letter { font-size : 300%; font-weight : bold; float : left;
 margin-right: 3px; }
```

then add following code to file */wp-includes/js/quicktags.js* and put them among *edButtons*:

```
 edButtons[edButtons.length] = new edButton('ed_capdrop' ,'CapDrop' ,'<
p id="fp">' ,'</p>' ,'c' );
```

You will now see a CapDrop quicktag in your edit window.

See also:

- drop caps support question

Where should I look for more information on CSS?

See CSS

Text and Content Display

How do I show only the titles of articles on the weblog homepage?

To show only the title of posts on the weblog homepage, in your theme's *index.php* replace

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

With something like:

```
<?phpif (is_single()) {the_content();}else {//no content, nothing.}?>
```

This will show the post content only on the individual posts page. Note if your theme uses another template, such as *single.php*, to display individual posts, then this change may not affect your individual post pages.

See also:

- Template Hierarchy
- Stepping Into Template Tags
- Stepping Into Templates

How do I show an excerpt, teaser, or summary, for an article on the blog homepage?

A **teaser** should not be confused with the **excerpt**. A teaser refers to the first few sentences or paragraphs of a post. When typing a long post, you can insert the <!--more--> Quicktag after a few sentences, and that acts as a cut-off point for the teaser. When the post is displayed on a home page, category page, or an archive page, the teaser is displayed, followed by a hyperlink (such as **Read the rest of this entry...**). Visitors can click on that link to see the full version of your post.

Note that some Themes may not support the **more** ability. Additional information on how to present the **more** can be found in the the_content() and Customizing the Read More articles.

As stated before, the teaser (the more) feature should not be confused with the Excerpt field that is completed when writing or editing a post. The Template Tag, the_excerpt(), can be used to display the post's excerpt field.

The excerpt you enter when writing a post will not be displayed on your blog unless the_excerpt() template tag is specified in your theme. Also, if you choose the "Summary" option **For each article in a feed** in Administration> Settings> Reading, the excerpt will be used for feeds.

How do I show a summary rather than the full content of posts?

Here's two possible ways to show a summary, rather than full content of posts on a blog main page:

1. Use the Quicktag *<!--more-->* in your posts, and it will display the text of the post up to that point, and then provide a link, such as "Continue reading...", to allow the reader to see the complete post. The article, Customizing the Read More, discusses changing the text (e.g. Continue reading...) of the link.
2. Change your theme's index.php Template to use the Template Tag, the_excerpt(), instead of the_content().

See also:

- Stepping Into Template Tags
- Stepping Into Templates
- Template Hierarchy

How do I customise the text shown in "(more.....)" on my weblog?

See Customizing the Read More

How do I use the blogroll links rating feature to display the ratings?

In WordPress 1.5 and later, you have the option to rate the links in your Link Manager (Blogroll) and display the ratings for the world to see.

To rate your links, edit the particular link (by using the "Links" editing interface, or "Blogroll" in later versions of WordPress -- rating is in the Advanced section), and rate the link from 0 - 10 using the dropdown menu.

To get your link ratings to display, you may need to edit your Theme. See the documentation for the wp_list_bookmarks() or get_links() Template Tags for more information.

In WordPress 1.2, the procedure is slightly different. First, for each link category, you have to turn on the display of link ratings (Links -> Link Categories -> Show -> Rating).

Once that is done, you can display the ratings by changing the settings in the Options -> Link Manager screen:

- a character (* by default)
- a number (the rating, 0-9)
- images -- if you are using Images, you have the choice of using the same image displayed n times to show a rating of n, or using 10 different images, for one through 10.

Images and Graphics

How to add a favicon to your site

To add a favicon to your site in WordPress 2.0, place your **favicon.ico** file inside your theme folder (for example: *wp-content/themes/default/*) then add this line to *header.php*:

<link rel="shortcut icon" href="<?php bloginfo('template_directory'); ?>/favicon.ico" />

Be sure to add it somewhere within the <head></head> section.

See Creating and Installing a Favicon For more detailed instructions.

Where can I get some buttons for my site?

Buttons are like badges you display on your website to show your affection for something, or to display information regarding your cultural, social, political or technical leanings.

To add a WordPress button to your site showing support for the WordPress Community:

- Contributing to WordPress, Show you Care

For more buttons, see:

- GtmcKnight's Steal these Buttons
- Adam Kalsey's Buttonmaker

How do I get WordPress to generate links to the thumbnail of an uploaded picture?

WordPress by default only produces very limited code after uploading a picture. If you want to get code for the thumbnail link, the thumbnail etc, autogenerated, the following resources will be useful, including a possible hack that may not work for WordPress v1.5 at Thumbnail HTML addition for upload.php.

See also:

- Using Images
- Photoblogs and Galleries

How can I use a custom image for my list bullets?

See also:

- Styling Lists with CSS
- Customizing Your Sidebar

How do I rotate/cycle/randomize the image in the header of the weblog?

See also:

- Designing Headers, Rotating Header Images

How do I link to my own images?

You can use absolute or relative URI/URLs addresses.

To use a relative link, set the address from the **root** folder of your site by using a slash in front of the folder in the root directory.

```
<img src="/images/balloons/image.jpg" alt="balloons" />
```

To use an absolute link:

```
<img src="http://www.example.com/images/balloons/image.jpg" alt="balloo
ns" />
```

How can I display images in my category and archive pages?

When using the default theme, you'll notice images (and links) do not appear when visiting category and archive query pages. This has to do with how the default theme displays post

content in those sections of your blog. To change this behavior, edit the default theme's Archive Template (archive.php). You can do this online through the Theme Editor, or offline by downloading and opening the default theme's archive.php in any text editor. Once in the Archive Template, look for this section:

```
<div class="entry"><?php the_excerpt() ?></div>
```

Here, change the_excerpt() template tag, which displays a summary of a post's content while filtering out all HTML tags. To display each post's whole content (and HTML tags), use the_content() template tag:

```
<div class="entry"><?php the_content(); ?></div>
```

Where can I find more information on images, PhotoBlogs, and photo galleries?

See also:

- Using Images
- Photoblogs and Galleries

Template Tags

How do I change the time stamp for each post from an AM/PM mode to a 24 hour mode?

You set the **Default Time Format** for your blog via the Administration> Settings> General under theDate and Time section.

See also:

- Formatting Date and Time

How can I have the date/time displayed on every entry I make?

To put the date and time on every post title on your site, you may have to change more than one template file. They may include `index.php`, `single.php`, `category.php`, and `archives.php`.

From among the various template files, find all references to the title of your post like this (your Theme version may be slightly different):

```
<h2><a href="<?php the_permalink() ?>" rel="bookmark" title="Permanent
Link to <?php the_title(); ?>"><?php the_title(); ?></a></h2><small><?p
hp the_time('F jS, Y') ?> by <?php the_author() ?></small>
```

Rearrange it so the time information goes in front (or in back) of your Post Title:

```
<h2><a href="<?php the_permalink() ?>" rel="bookmark" title="Permanent
Link to <?php the_title(); ?>"><?php the_time('F jS, Y') ?> - <?php the
_title(); ?></a></h2><small>by <?php the_author() ?></small>
```

See also:

- Formatting Date and Time

How do I change the "Permanent Link to" in my Title link?

The `title` of your links includes text that explains what the link is to, in concordance with web accessibility standards. By default, your title may look like this example, which uses the `title` attribute with the words "Permanent Link to" and the template tag that displays the title of the post.

```
<h2><a href="<?php the_permalink() ?>" rel="bookmark" title="Permanent
Link to <?php the_title(); ?>"><?php the_title(); ?></a></h2>
```

To change the "Permanent Link to" text, simply delete it and replace it with your own words:

```
<h2><a href="<?php the_permalink() ?>" rel="bookmark" title="Post about
 <?php the_title(); ?>"><?php the_title(); ?></a></h2>
```

Or remove it completely, leaving only the title tag.

```
<h2><a href="<?php the_permalink() ?>" rel="bookmark" title="<?php the_
title(); ?>"><?php the_title(); ?></a></h2>
```

How do I make my Categories appear in alphabetical order?

In some cases it may be necessary to change *index.php*.

Find this line:

```
<?php list_cats(0, 'All', 'name'); ?>
```

and replace it with this line instead:

```
<?php list_cats(0,'','name','','',true,0,1,1,1); ?>
```

See also:

- Template Tags/wp_list_cats and Template Tags/list_cats

How do I make my categories appear in a drop down list ?

In some cases it may be necessary to change *index.php*.

Find this line:

```
<?php list_cats(0, 'All', 'name'); ?>
```

and replace it with this line instead:

```
<form action="<?php echo $PHP_SELF ?>" method="get"><?php dropdown_cats
(); ?><input type="submit" name="submit" value="view" /></form>
```

See also:

- Template_Tags/dropdown_cats

How do I exclude one or more categories from being listed in the list of categories?

Use the following function to list your categories but exclude category 1:

```
<?php wp_list_cats('exclude=1'); ?>
```

Of course, change the **1** to the ID of the category you want to exclude.

To exclude multiple categories, use this:

```
<?php wp_list_cats('exclude=1, 2'); ?>
```

Change 1 and 2 to the categories you want excluded. You can exclude more of them by adding their IDs separated by commas.

See also:

- Template Tags/wp_list_cats and Template Tags/list_cats

How do I hide posts belonging to a certain category on the front page *index.php*?

If you need to hide (exclude) posts belonging to a certain category from displaying on the front page, you can place code that does the exclusion inside The Loop of your theme's `index.php` file.

The Loop starts something like this:

```
<?php if (have_posts()) : while (have_posts()) : the_post(); ?>
```

To exclude category 4 from the front page, just inside The Loop, add this condition :

```
<?php if ( !(in_category('4')) || !is_home() ) { ?><!-- Output the post
 here -->
```

The Loop ends something like this:

```
<?php endwhile; ?>
```

Just before that line, add this:

```
<?php } ?>
```

In the end, it will look like:

```
<?php if (have_posts()) : while (have_posts()) : the_post(); ?><?php if
 ( !(in_category('4')) || !is_home() ) { ?>

<!-- Output the post here -->

<?php } ?><?php endwhile; ?>
```

This means that if on the front page, the post will be presented if it's not in category 4. On pages other than the front (home) page, all posts are presented.

See also:

- Exclude Posts From Some Category

How do I make my Archives appear in a drop-down list?

Put this code into your *index.php* where you wish the item to appear:

```
<li id="archives">Archives:<ul><li><form name="archiveform" action=""><
select name="archive_chrono" onchange="window.location =  (document.for
ms.archiveform.archive_chrono[document.forms.archiveform.archive_chrono
.selectedIndex].value);"><option value=''>By Month</option>< ?php get_a
rchives('','','option', 1); ?></></select></form></li></ul></li>
```

How do I get rid of the "No Comments" message displayed with every post?

If you do not allow comments on your site you may want to get rid of the "No Comments" (or Comments are off) message displayed with each post.

If you are using the WordPress Default theme you would delete the following code from `wp-content/themes/default/index.php`

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

Note that if you use a different theme the information to delete may be slightly different.

See also:

- comments_popup_link()

Templates

Why can I see only n posts on the blog, where are the other entries?

Make sure that you have the correct settings in the **Show at most** posts or days fields in the Administration> Settings> Reading Panel.

If you are not seeing all your entries and you modified the default index.php, make sure you have an equal number of opening and closing tags, and that they are in the right places.

How do I create an archives page, with all the entries sortable by different methods?

See also:

- Creating an Archive Index
- Plugins
- Another Nicer Archives Version
- Sporadic Nonsense's Clean Archives Plugin

What do the _() and _e() functions in WordPress do?

In the simplest of terms, they "print" what you tell them to do. They are abbreviations for the PHP term "echo" which displays text. In WordPress, they are used to identify strings in the php files marked for translation to other languages, and localization using two "tags" which are actually functions. They are:

- _()
- _e()

These accept a string as an argument. For example:

```
_("Translate Me")_e("Translate Me")
```

The only functional difference between the two methods is that _e() echoes the string and _() simply returns the string. _() is used when you want to supply a string to a function. _e() is used when you want to output the string as part of your XHTML.

We have a tool which goes through all of the php files, extracting strings that are marked by **_()** and **_e()**.

See also:

- Conditional Tags

How to easily display links to both Pages and Categories in the blog navigation header?

This discussion assumes your current navigation is (or will be) using the Template Tag, wp_list_pages(), to display links to your Pages. The trick to displaying both Pages and Categories in a blog navigation header is to make the Categories accessible via a Page.

To display a link to your News Category along with the other Pages in your navigation header, install a plugin such as Page Links To, create a Page called News, then in the Page Links To module, in the "Point to this URL:" field, enter the **URL for the News category**. Note: to determine the URL for the News Category, just visit that Category in your blog and copy the URL in your browser address bar.

If you want the Pages in your navigation to appear in a particular order, use the **Order** field, in the Attributes module when you edit your Page and set that correctly for each Page, then with wp_list_pages(), use the **'sort_column=menu_order'** parameter.

Other plugins with similar solution to Page Links To:

- http://svn.redalt.com/projects/Redirectify/trunk/redirectify.php
- http://wordpress.org/extend/plugins/redirect/

See also:

- Managing Plugins

How to display posts in a Page?

There's several ways to do display posts in a Page. The simplest way is described in the FAQ: How can I have a static front page and posts display on a page called Blog?

A more complex method is to first create a Page Template with a query, and a loop, that retrieve and display the posts. Then via Administration> Pages> Add New, add a new Page with that Page Template designated in the Template field. Here's an example developed using the WordPress Default theme:

- 1. Create a file called *wp-content/themes/default/pageofposts.php* that has this code:

```
<?php/*Template Name: PageOfPosts*/get_header(); ?><div id="content" cl
ass="narrowcolumn"><?php$showposts = -1; // -1 shows all posts$do_not_s
how_stickies = 1; // 0 to show stickies   $args=array(   'showposts' =>
 $showposts,   'caller_get_posts' => $do_not_show_stickies,   );$my_que
```

```
ry = new WP_Query($args); ?><?php if( $my_query->have_posts() ) : ?><?p
hp while ($my_query->have_posts()) : $my_query->the_post(); ?><?php//ne
cessary to show the tags global $wp_query;$wp_query->in_the_loop = true
;?><div <?php post_class() ?> id="post-<?php the_ID(); ?>"><h2><a href=
"<?php the_permalink() ?>" rel="bookmark" title="Permanent Link to <?ph
p the_title_attribute(); ?>"><?php the_title(); ?></a></h2><small><?php
 the_time('F jS, Y') ?><!-- by <?php the_author() ?> --></small><div cl
ass="entry"><?php the_content('Read the rest of this entry »'); ?></div
><p class="postmetadata"><?php the_tags('Tags: ', ', ', '<br />'); ?> P
osted in <?php the_category(', ') ?> | <?php edit_post_link('Edit', '',
 ' | '); ?><?php comments_popup_link('No Comments »', '1 Comment »', '%
 Comments »'); ?></p></div><?php endwhile; ?><div class="navigation"><d
iv class="alignleft"><?php next_posts_link('« Older Entries') ?></div><
div class="alignright"><?php previous_posts_link('Newer Entries »') ?><
/div></div><?php else : ?><h2 class="center">Not Found</h2><p class="ce
nter">Sorry, but you are looking for something that isn't here.</p><?ph
p get_search_form(); ?><?php endif; ?></div><?php get_sidebar(); ?><?ph
p get_footer(); ?>
```

- 2. In Administration> Pages> Add New, create a new Page, title the Page whatever you want, and in the Template field select **PageOfPosts**.
- 3. That's it, now visit that Page on your blog and you should see your posts.

See also:

- Why is there no Page Template option when writing or editing a Page?

How to delete duplicate navigation bar references to Home?

If your theme displays Pages for navigation, and there is more than one instance of Home in the navigation bar, here's several ways to fix that. Typically the duplicate problem occurs after someone has created a Page called Home.

Many theme authors 'hard-code' a reference to Home in the theme's **header.php** and use template tag, >wp_list_pages, to display all the other Pages. But, if you've created a Page called Home, that likely causes the duplicate navigation item.

To delete the duplication, you would edit your theme's **header.php** and:

1. Find and delete the hard-coded reference to Home.
2. Or, use the **exclude=x** argument with wp_list_pages() and replace the **x** with the Page ID of your Home Page.

How to get rid of encoding in a theme's footer?

Some theme authors embed links in an encoded footer making it difficult to understand what is happening in the footer. **Note:** this type of encoded content could be malicious as well as just a copyright banner! Here's a method to figure out and correct the footer so there is no encoding.

In your index.php find the line that says *<?php get_footer(); ?>*. Above and below it add marker text like this:

```
<!-- Evil Footer Devil FOUND --><?php get_footer(); ?><!-- Evil Footer
Devil BEGONE -->
```

Now visit your blog, view source (e.g. View->Page Source in Firefox), and copy the HTML between those two markers. Rename your footer.php and make a new copy of *footer.php* with

that HTML code. Then change it to your hearts content. Remember to insert the *<?php wp_footer(); ?>* before *</body>* tag.

Keep in mind that if your theme's license does not permit this, then don't do it. If that's the case please consider finding a new theme.

See also:

- http://wordpress.org/support/topic/237083
- http://wordpress.org/support/topic/235287

Themes

How do I use a Theme style from Alex King's site?

See also:

- Using Themes
- Using Themes/Theme List
- Alex Kings Theme Competition

Which files do I modify when I start to design my site?

See also:

- Working with WordPress, Site Development

How do I tell which file is making which output?

You can put e.g.,

```
<!-- Begin <?php echo basename(__FILE__); ?> --><!-- End   <?php echo b
asename(__FILE__); ?> -->
```

at the top and bottom of your files, which will be printed in the HTML output.

How do you create a screenshot.png for your Theme?

See also:

- Tanster's Quick and dirty way to create screenshot.png

Back to FAQ

HTML to XHTML

What is and what isn't XHTML

WordPress, as a system, is based on documents written in the XHTML scripting language. XHTML 1.0 (which is currently the most widely supported version) became a W3C recommendation in the year 2000, and was intended to serve as an interim technology until XHTML 2.0 could be finalized. Eight years later XHTML 2.0 still isn't finished. This document therefore uses the phrase XHTML to refer to XHTML 1.0 only.

XHTML is very similar to HTML as both are descended from a language called SGML. However, XHTML is also descended from XML, which is a scripting language with much stricter grammar rules than HTML, and XHTML has inherited some of that discipline. XHTML is mainly differentiated from HTML by its use of a new MIME TYPE and the addition of some new syntax rules which are explained below.

Why Should I use XHTML

WordPress prints XHTML from all it's internal functions, all themes therefore are now in XHTML and so too are most plugins. So, if you want to use WordPress, you should buckle down and learn some XHTML as that's where it is right now.

What are the differences between XHTML and HTML

If you are familiar with HTML, you will be glad to know that the majority of what you know about HTML is still relevant in XHTML. The main differrences are that XHTML forces webpage authors to be more consistent and to write more legible code. There are a few syntax and grammar differences and a few HTML tags have been dropped and, really, that's about it. If you know HTML then you'll be surprised at how easy it is to switch to XHTML, and the new XHTML rules will force you to become a better programmer too!

So how do I write XHTML?

Well, here's a quick check list of the important requirements of XHTML and the differences between it and HTML. This is NOT a comprehensive XHTML language reference. Please look here for that.

In these examples some code has been omitted for clarity

All tags, attributes and values must be written in lowercase.

Right:

```
<a href="www.kilroyjames.co.uk">
```

Wrong:

```
<A HREF="www.kilroyjames.co.uk">
```

All attribute values must be within quotes

Right:

```
<a href="www.kilroyjames.co.uk">
```

Wrong:

```
<a href=www.kilroyjames.co.uk >
```

All tags must be properly nested

Right:

```
<em>this emphasis just keeps getting <strong>stronger and stronger</str
ong></em>
```

Wrong:

```
<em>this emphasis just keeps getting <strong>stronger and stronger</em>
</strong>
```

All XHTML documents must carry a DOCTYPE definition

The DOCTYPE is an intimidating looking piece of code that must appear at the start of every XHTML document, it tells the browser how to render the document.

Rules for the DOCTYPE tag:

- It must be the first tag in the document
- The DOCTYPE is not actually part of the XHTML document so don't add a closing slash
- It should point to a valid definition file called a DTD that tells the browser how to read the document
- You must write the DOCTYPE tag correctly otherwise your document will explode* (into little pieces of HTML called "tag soup") and be unvalidateable.

* I am, of course, perfectly serious...

There are three types of valid XHTML 1.0 document: Strict, Transitional, and Frameset. If you can get your document to validate with "Strict" then do so, however some legacy tags and attributes aren't allowed in Strict so you can use "Transitional" instead.

Note: you might have a problem getting WordPress to validate as Strict because, as of version 2.6.2, some of the internally generated <form> elements still use the "name" attribute, which is not allowed under the Strict DTD, ie. <input name="my_button" />

Note also that using a Transitional DTD takes most browsers out of "Standards" mode. It is

much trickier to get your web pages to look consistent across different browsers when the browsers are not in Standards mode. I'm not going to explain the minutae of the DOCTYPE tag as it gets deeper and more complicated, just know that for best results you should use one of the following, preferably the first one (Strict):

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

The HTML tag must contain an XMLNS attribute

You don't need to understand the "XML namespace" attribute, except to know that it is required in all XHTML documents. Here is an example of how to write it:

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

Documents must be properly formed with HTML, HEAD, TITLE and BODY tags

In HTML it is possible to write a webpage that contains none of the above tags; in XHTML it is not. The above tags must be included and they must be nested and ordered correctly, as follows (the DOCTYPE has been omitted):

```
<html xmlns="http://www.w3.org/1999/xhtml"><head><title></title></head>
<body><p>       See how the TITLE must be placed in the document HEAD —
 the TITLE is considered
to be a "required child" element of the HEAD.
 Notice that the HEAD must also appear before the document BODY.
Notice also how both the HEAD and the BODY must be contained
 within the HTML tag. Again, HEAD and BODY are "required child"
 elements of the HTML tag. Finally, notice that this text is
written within a <p>paragraph</p> tag; in XHTML you may
not write text directly in the BODY tag without using a suitable
container tag, such as <p> or <div>.  </p></body></html>
```

All tags must be closed, even single tags

```
<p>Mary had a little lamb<p>It's fleece was white as snow
```

This code is not valid XHTML as the closing </p> tags have been left out. The following example is correct.

```
<p>Mary had a little lamb</p><p>It's fleece was white as snow</p>
```

In XHTML even single tags have to be closed - absolutely NO tag may be left open.

```
<p> Mary had a little lamb <br> It's fleece was white as snow</p>
```

Therefore the above example is wrong because the
 tag is not closed. To close a single tag like
 and <hr> you simply add a forward slash before the final bracket, like so:

and <hr /> (the white space is optional). To correct the above example we'd write:

```
<p> Mary had a little lamb <br/> It's fleece was white as snow</p>
```

This is correct XHTML.

Attribute minimisation isn't allowed

In HTML, attributes can be strung together almost like they were keywords, ie. <dl compact>, this is called attribute minimisation.In XHTML that is not allowed, attributes and values must be excplicit, ie.

```
<dl compact="compact">
```

ID and NAME attributes

In HTML it was legal to use ID and NAME attributes interchangeably. In XHTML the NAME attribute is formally deprecated and cannot be used. In all cases where you would think to use a NAME attribute you must now use ID instead. e.g.

correct HTML

```
<input type="submit" name="s" value=" Search ">
```

and now the correct XHTML version

```
<input type="submit" id="s" value=" Search " />
```

STYLE is all in your HEAD

XHTML does not allow STYLE declarations within the body of a document they must be placed in the document HEAD instead.

Entity references

Write all literal ampersands as & or they will be assumed to be part of an entity reference. e.g. ® is the entity reference for the symbol ® Therefore M&S is invalid XHTML because &S is not an entity reference, you must write it as M&S.

Conclusion

As I said, this is not a comprehensive reference but it should be enough to get you up and running with XHTML pretty quickly. Good luck!

Problems with XHTML

Most people don't realise that to use XHTML properly it must be served using the new MIME TYPE "application/xhtml+xml". A MIME TYPE is simply a description that the web server sends to a browser to tell it what sort of document is coming. For instance a JPG image is sent with a MIME TYPE of "image/jpeg" and an HTML document with a MIME TYPE of "text/html".Sending

an XHTML document with a MIME TYPE of "text/html" results in the document being parsed and validated as HTML, not as you would no doubt hope, as XHTML. You must use the correct MIME TYPE if you want to use XHTML otherwise you are simply using non-standard HTML. In order to avoid this problem and output standards compliant code you can use the XHTML to HTML wordpress plugin.

HTML 5

Because of seemingly intractable problems with the development of XHTML (mainly that XHTML 2 is incompatible with previous versions of XHTML and HTML and also the MIME TYPE issue), a competing standard supported by Mozilla (Firefox), Apple (Safari), Opera, Microsoft (Internet Explorer) and some other key Internet players has become the new favourite to succeed the old HTML 4.01 standard.

HTML 5 was passed as a draft recommendation by the W3C in January 2008 and is expected to become a full recommendation within the next couple of years.

Resources

- W3C XHTML language specification
- The W3Cs Markup Validation Service (use it!)
- Wikipedia: XHTML
- Wikipedia - Common XHTML errors
- Wikipedia: HTML 5
- Beware of XHTML

Business Blogging and Shopping

A recent article How to Turn a Normal WordPress Installation into a Working Online-Shop discusses how to turn a normal WordPress installation into a working online shop. Although the article is very thorough and well written, it is not the easiest method to turn your WordPress installation into a working online shop. The easiest way to achieve this is by downloading and installing the WP e-Commerce plugin.

e-Commerce Install

Assuming you have a working version of WordPress installed you must do the following:

1. Download the e-Commerce plugin
2. Copy the folder to your local plugins directory
3. Activate the plugin and configure it to your liking

e-Commerce setup

Once the plugin is installed the most important things to do are:

1. Set your default shipping country and associated shipping settings
2. Setup and configure the presentation settings
3. Setup your receipt notification email address
4. Setup your payment gateway
5. Add products
6. Upgrade to gold if you want to add multiple images to products

PayPal Setup

These are the instructions for setting up PayPal for your WP e-Commerce store.

1. Browse to PayPal and sign up using your email address
2. Under Profile > Shipping Settings check the allow transaction-based shipping values to override the profile shipping settings listed above

Google Checkout Setup

These are the instructions for setting up Google Checkout for your WP e-Commerce store

1. Browse to Google Checkout and click sign up now
2. Select whether or not your business has a Google Account for services like AdWords or Gmail?
3. Create your account using your Gmail email address

At your WP e-Commerce store's Admin

Now that you have a Google Checkout account setup enter your Google Checkout Merchant ID and Key into the payment gateway settings page in the WP e-commerce admin panel

1. Under e-Commerce > Gateway Options choose Google Checkout as your payment gateway.
2. In "Google Checkout Merchant ID" enter your Merchant ID.
3. In "Google Checkout Merchant Key" enter your Merchant Key.
4. Choose your Server Type (Sandbox vs Production), currency, and button style
5. Click the button that says "Submit"

Conclusion

Now that you have setup your shop and have a working PayPal account you are ready to start selling! Upload your products and have fun.

For more detailed information on how to use WP e-Commerce you can read (or contribute) to the new WP e-Commerce project Wiki.

Business Blogging

Why stop there? Business blogging is really starting to take off. More and more people are starting to use WordPress to power their websites - especially now that there are so many powerful plugins. Lets take a quick glance at some of the plugins we use to turn your average WordPress install into a business blogging solution.

- Static Front Page. You might not want the front page of your web site to be your blog. See Creating a Static Front Page.
- Forums. Use bbPress or WP Forum depending on the requirements. Both are easy to install and both have a range of useful features.
- Email Newsletters. Use WP Campaign Monitor because it is simple to install and easy to get the hang of. It also integrates with WP e-Commerce lite.
- Surveys. Information is power. Use the Survey Fly plugin because it is powerful and easy to use. Administrators can download .csv reports that they can manipulate internally using proper spreadsheet software.
- Statistics. Use wp-slimstat or wp-slimstat-ex because they are both so easy to install. WP slimstat ex is based on wp-slimstat but full of AJAX goodies.
- Google Analytics. Google Analytics tells you everything you want to know about how your visitors found you and how they interact with your site. The Google Analytics Plugin for WordPress adds an analytics panel to the WordPress admin website.
- Search Engine Optimization. Search Everything enables the searching of Pages, Comments and more. No Hacking or modifications necessary, just install, activate the plugin and configure it under Options > Search Everything
- Ecommerce & Shopping Cart. Turn your Wordpress Installation into a fully functional Ecommerce Platform. See the available Ecommerce Plugins & Ecommerce Themes for Wordpress.

CSS

Languages: **English** • 日本語 • (Add your language)

WordPress relies heavily on the presentation styles within CSS. With the introduction of WordPress v1.5 Themes, your layout options haven't just expanded, they've exploded! WordPress has made it easier than ever to change your website look, and opened up the field even more to help you create your own Theme and page layout.

CSS stands for **Cascading Style Sheets**. It allows you to store style presentation information (like colors and layout) separate from your HTML structure. This allows precision control of your website layout and makes your pages faster and easier to update.

This article briefly describes the use of CSS in WordPress, and lists some references for further information. For information on CSS itself, see Know Your Sources#CSS.

WordPress and CSS

WordPress Themes use a combination of template files, template tags, and CSS style sheets to generate your WordPress site's look.

Template Files
: Template files are the building blocks which come together to create your site. In the WordPress Theme structure, the header, sidebar, content, and footer are all contained within individual files. They join together to create your page. This allows you to customize the building blocks. For example, in the Default WordPress Theme, the multi-post view found on the front page, category, archives, and search web pages on your site, the sidebar is present. Click on any post, you will be taken to the single post view and the sidebar will now be gone. You can choose which parts and pieces appear on your page, and customize them individually, allowing for a different header or sidebar to appear on all pages within a specific category. And more. For a more extensive introduction to Templates, see Stepping Into Templates.

Template Tags
: Template tags are the bits of code which provide instructions and requests for information stored within the WordPress database. Some of these are highly configurable, allowing you to customize the date, time, lists, and other elements displayed on your website. You can learn more about template tags in Stepping Into Template Tags.

CSS Style Sheets
: This is where it all comes together. On every template file within your site, there are XHTML tags and CSS references wrapped around your template tags and content. In the style sheet within each Theme are commands for the page's structure. Without these instructions, your page would simply look like a long typed page. With these instructions, you can move the building block structures around, making your header very long and filled with graphics or photographs, or simple and narrow. Your site can "float" in the middle of the viewer's screen with space on the left and right, or stretch across the screen, filling the whole page. Your sidebar can be on the right or left, or even start midway down the page. How you style your page is up to you. But the instructions for styling are found in the `style.css` file within each Theme folder.

WordPress Generated Classes

Several classes for aligning images and block elements (DIV, P, TABLE etc.) were introduced in WordPress 2.5: `aligncenter`, `alignleft` and `alignright`. In addition the class `alignnone` is added to images that are not aligned, so they can be styled differently if needed.

The same classes are used to align images that have a caption (as of WordPress 2.6). Three additional CSS classes are needed for the captions, together the alignment and caption classes are:

```
.aligncenter,div.aligncenter {   display: block;   margin-left: auto;
 margin-right: auto;}.alignleft {   float: left;}.alignright {   float:
 right;}.wp-caption {   border: 1px solid #ddd;   text-align: center;
 background-color: #f3f3f3;   padding-top: 4px;   margin: 10px;   /* op
tional rounded corners for browsers that support it */   -moz-border-ra
dius: 3px;   -khtml-border-radius: 3px;   -webkit-border-radius: 3px;
 border-radius: 3px;}.wp-caption img {   margin: 0;   padding: 0;   bor
der: 0 none;}.wp-caption p.wp-caption-text {   font-size: 11px;   line-
height: 17px;   padding: 0 4px 5px;   margin: 0;}
```

Each theme should have these or similar styles in its `style.css` file to be able to display images and captions properly.

Additionally, there are a few more WordPress class tags that you may optionally wish to style because they are generated by default:

```
.categories {...}.cat-item {...}.current-cat {...}.current-cat-parent {
...}.pagenav {...}.page_item {...}.current_page_item {...}.current_page
_parent {...}.widget {...}.widget_text {...}.blogroll {...}.linkcat{...
}
```

See also

To help you understand more about how CSS works in relationship to your web page, you may wish to read some of the articles cited in these lists:

- Templates - Comprehensive list of WordPress Theme and Template articles (a good starting point is Using Themes, and there are also many advanced articles in this list)
- Blog Design and Layout - Comprehensive list of resources related to site design in WordPress
- WordPress Lessons - Lessons on all aspects of WordPress

WordPress Layout Help

If you are having some problems or questions about your WordPress Theme or layout, begin by checking the website of the Theme author to see if there is an upgrade or answers to your questions. Here are some other resources:

- Lessons on Designing Your WordPress Site

- Site Architecture 1.5
- CSS Troubleshooting
- Finding Your CSS Styles
- CSS Fixing Browser Bugs
- Validating a Website
- FAQ Layout and Design
- Templates
- WordPress Support Forums
- WordPress CSS Guides by Podz
- CSS Shorthand
- HTML to XHTML NEW!

Finding Your CSS Styles

The following isn't about choosing a WordPress Theme, but finding the CSS styles within your current Theme. Often a problem occurs in one part of the layout or template. For instance, there is a border you don't want between the sidebar menu and the rest of the page. You hunt and hunt but you can't find any reference to the border. Now what?

Playing CSS Detective

Let's begin by playing CSS detective. You know where the problem is, you just can't *find* the problem. In the above example, you need to hunt for an errant border.

Begin by carefully examining a generated page (or test page) and look for some identifying text in the sidebar, near the errant border. Let's say that listed in the sidebar, you have a post title called "All About Harry". You know you'll find that title in your sidebar when you view the page's source code.

To view a page's source code, go up to the **menu bar** of your browser and choose **VIEW > PAGE SOURCE or VIEW > SOURCE**. A page will pop up featuring the source code of the page.

Use your handy detective tool, **Ctrl+F**, to activate your search. Type in "all about harry" and click **FIND**. Odds are, unless you have the words "all about harry" in your post, it will take you to the first showing of the phrase "all about harry" which is probably in your sidebar. If not, hit FIND again until you've found the right phrase in the right area.

If you are using Internet Explorer, an alternate method is to use the Internet Explorer Developer Toolbar, which allows you to visually see and select the elements, IDs, and classes on the page. It displays the elements within the hierarchy of the page, their CSS attributes, and can outline DIVs, tables, etc. You can download the Toolbar from Microsoft.

Once you've found the phrase, it's time to play CSS detective. Look up through the code from the phrase "All About Harry" for one of two things. It will look something like either of these, using words like **sidebar**, **menu**, or **sidecolumn**:

```
<div id="sidebar">or<div class="sidebar">
```

This is the main section that contains your sidebar menu. You've found the first suspect.

Now, open your `style.css` file and do another search for **sidebar** or whatever the resulting name was that you uncovered. It is usually identified in two ways:

```
#sidebaror.sidebar
```

Look in the styles under these CSS **selectors** and see if there is a mention of border, often looking something like this:

```
#sidebar {position: relative; float: right; width: 170px;    color: blu
e; font-size: 90%; border-right: solid 1px blue; }
```

There is your border, the criminal! If this is the guilty party, delete the reference to the border and you are good to go.

If it isn't, the hunt continues.

Sometimes the culprit is the one you least suspect. Maybe the border is not caused by the obvious suspect, the *sidebar*, but by the **content** section. Return to the generated page source code and search for the first words of your post. Look above that for something like:

```
<div id="content">
```

It could be called **content**, **page**. **post**, **maincolumn**, **widecolumn**, or have another alias, but it should be the CSS *container* that holds your post information. Now, go back to the style sheet and check to see if there is a border in that section.

Frisk the Style Sheet

If all of these fail, the CSS detective never gives up the hunt. Return to the hiding place of all styles, the `style.css` file, and *frisk it* by doing a search for "border" and look carefully at each suspect. Note the selector ID name, like **sidebar**, **menu**, **content** and **page**, and then go back to the generated page source to see if that might be your culprit.

You can also select the border suspect you've found on the style sheet and **cut and paste** it into a TXT file (like Notepad) that just sits open on your computer like a scratch notepad. Make a note of which selector name you removed it from like this:

```
Removed border: solid 2px green from #content
```

Then save the edited `style.css` and upload it to your site. Refresh the generated test post and see if the unwanted border is gone. If so, you found the culprit. If not, return to the Notepad and copy the code and put it back into your `style.css` in the "content" section, putting things back where you found it.

If you do find your culprit, do a little dance, squeal and cheer, and make others suspicious and nervous when they are around you. The CSS detective solves another CSS crime!

More CSS Troubleshooting Help

- CSS Fixing Browser Bugs
- CSS Troubleshooting

Pages

Languages: **English** • 日本語 • ไทย • 简体中文 • (Add your language)

In WordPress, you can write either posts or pages. When you're writing a regular blog entry, you write a post. Posts automatically appear in reverse chronological order on your blog's home page. Pages, on the other hand, are for content such as "About Me,""Contact Me," etc. Pages live outside of the normal blog chronology, and are often used to present information about yourself or your site that is somehow timeless -- information that is always applicable. You can use Pages to organize and manage any amount of content.

Other examples of common pages include Copyright, Legal Information, Reprint Permissions, Company Information, and Accessibility Statement. (By the way, it's a good idea to always have an about page and a contact page -- see this advice from Lorelle.)

In general, Pages are very similar to Posts in that they both have Titles and Content and can use your site's Presentation Templates to maintain a consistent look throughout your site. Pages, though, have several key distinctions that make them quite different from Posts.

Pages in a Nutshell

What Pages Are:

- Pages are for content that is less time-dependent than Posts.
- Pages can be organized into pages and SubPages.
- Pages can use different Page Templates which can include Template Files, Template Tags and other PHP code.

What Pages are Not:

- Pages are not Posts, nor are they excerpted from larger works of fiction. They do not cycle through your blog's main page. (**Note:** You can include Posts in Pages by using the Inline Posts Plugin.)
- Pages cannot be associated with Categories and cannot be assigned Tags. The organizational structure for Pages comes only from their hierarchical interrelationships, and not from Tags or Categories.
- Pages are not files. They are stored in your database just like Posts are.
- Although you can put Template Tags and PHP code into a Page Template, you cannot put these into the content of a Page and expect them to run. (**Note**: You can achieve this by using a PHP evaluating Plugin such as Exec-PHP or RunPHP.)

Creating Pages

To create a new Page, log in to your WordPress installation with sufficient admin privileges to create new articles. Select the Administration> Pages> Add New option to begin writing a new

Page.

Changing the URL (or "Slug") of Your Pages

With 2.5, changing the page URL became less intuitive. If you have Permalinks enabled, and you have selected the **Day and Name** option (Click the **Settings** tab, and then click the **Permalinks** subtab), then the permalink automatically shows up below your post title when you start typing in the body of your post (not just the title).

However, if you have a different permalink option selected, or if you don't have permalinks enabled at all, you must do the following to edit your page URL:

1. Write a page by going to **Write > Page**.
2. Click the **Publish** button to publish your page.
3. Go to **Manage > Pages**.
4. Click **Edit** next to your page.
5. See the permalink under the title, and click the **Edit** link to change it.

Thus, if you don't have the right permalink option enabled, you have to publish your pages before you can set the URLs.

Listing Your Pages on Your Site

WordPress is able to *automatically* generate a list of Pages on your site, for example within the sidebar, using a Template Tag called `wp_list_pages()`. See the wp_list_pages page for information on how to do the following:

- Sort the list of Pages (to fully customize the order in which the Pages are listed, you might find the "Page Order" section on the Write > Page administration panel useful),
- exclude (or 'hide') a Page from the list,
- Control which Pages are displayed (i.e., all Pages or just certain SubPages), and
- Control how deep into your Page hierarchy the list goes.

Naturally, you can also link to Pages manually with an HTML link. For example, if you want your Copyright Page listed in your footer, that link might read as below:
If you *do not* have Permalinks set up

```
<a title="Copyright information" href="wordpress/?page_id=14">Copyright
 1996-2006</a>
```

If you *do* have Permalinks set up

```
<a title="Copyright information" href="wordpress/copyright/">Copyright
1996-2006</a>
```

Note: Your .htaccess file *must* be writeable for Page Permalinks to work, otherwise you must update your .htaccess file every time you create a Page.

Organizing Your Pages

Just as you can have Subcategories within your Categories, you can also have **SubPages** within your Pages, creating a hierarchy of pages.

For example, suppose you are creating a WordPress site for a travel agent and would like to create an individual Page for each continent and country to which the agency can make travel arrangements. You would begin by creating a Page called "Africa" on which you could describe general information about travel to Africa. Then you would create a series of Pages which would be SubPages to "Africa" and might include "Lesotho", "Cameroon", "Togo", and "Swaziland". Another individual Page is made for "South America" and would feature SubPages of "Brazil", "Argentina", and "Chile". Your site would then list:

- Africa
    - Cameroon
    - Lesotho
    - Swaziland
    - Togo
- South America
    - Argentina
    - Brazil
    - Chile

To begin the process, go to Administration> Write> Write Page panel, in the upper right corner of the panel and click the "Page Parent" drop-down menu. The drop-down menu contains a list of all the Pages already created for your site. To turn your current Page into a SubPage, or "Child" of the "Parent" Page, select the appropriate Page from the drop-down menu. If you specify a Parent other than "Main Page (no parent)" from the list, the Page you are now editing will be made a Child of that selected Page. When your Pages are listed, the Child Page will be nested under the Parent Page. The Permalinks of your Pages will also reflect this Page hierarchy.

In the above example, the Permalink for the Cameroon Page would be:

```
http://example.com/africa/cameroon/
```

Page Templates

Individual Pages can be set to use a specific custom **Page Template** (a PHP template file, e.g., snarfer.php) you create within your Theme (see Creating your own Page Templates below on how to create a custom template). This new Page Template will then override the default `page.php` Page Template included with your Theme. See What Template is used to Display a Particular Page? below, to find out exactly which Template will be used, but read the following first, so you understand the answer `:)`

WordPress can be configured to use **different Page Templates for different Pages.** Toward the bottom of the Write > Page administration panel (or on the sidebar, depending on which version of WordPress you are using) is a drop-down labeled "Page Template." From there you can select which Template will be used when displaying this particular Page.

NOTE: In order to access the Page Template selector, there must be at least one custom Page Template available in the active theme (see Creating your own Page Templates below to learn how to create one). If a custom page exists, but you still are not able to see Page Template

selector, try to re-activate your current theme.

Default Theme Page Templates

The Default theme contains three Page Templates for your use:

- `page.php` - Default Page Template: displays Page content
- `archives.php` - ignores Page content and instead displays a list of Archives by Month and Archives by Subject (by Category)
- `links.php` - ignores Page content and instead displays your links using wp_list_bookmarks()

What Template is used to Display a Particular Page?

WordPress will look for several template files in your active Theme. The first one it finds will be used to display any given Page. WordPress will look for files in the following order:

1. The Page's selected "Page Template"
2. `page.php`
3. `index.php`

Creating Your Own Page Templates

The files defining each Page Template are found in your Themes directory. To create a new Custom Page Template for a Page you must create a file. Let's call our first Page Template for our Page `snarfer.php`. At the top of the `snarfer.php` file, put the following:

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

The above code defines this `snarfer.php` file as the "Snarfer" Template. Naturally, "Snarfer" may be replaced with most any text to change the name of the Page Template. This Template Name will appear in the Theme Editor as the link to edit this file.

The file may be named *almost* anything with a `.php` extension (see reserved Theme filenames for filenames you should *not* use; these are special file names WordPress reserves for specific purposes).

What follows the above five lines of code is up to you. The rest of the code you write will control how Pages that use the Snarfer Page Template will display. See Template Tags for a description of the various WordPress Template functions you can use for this purpose. You may find it more convenient to copy some other Template (perhaps `page.php` or `index.php`) to `snarfer.php` and then add the above five lines of code to the beginning of the file. That way, you will only have to *alter* the HTML and PHP code, instead of creating it all from scratch. Examples are shown below. Once you have created the Page Template and placed it in your Theme's directory, it will be available as a choice when you create or edit a Page. (**Note**: when creating or editing a Page, the Page Template option does not appear unless there is at least one template defined in the above manner.)

Examples of Pages and Templates

The following is a list of instructional examples. Feel free to make additions.

Archives with Content

A Page Template that shows the Page's content at the top, and then displays a list of archive months and categories below it. This is designed to work with WordPress's Default theme (aka Kubrick), but will probably work with many other themes with a little modification.

Save this to `arc-cont.php`:

```
<?php/*Template Name: Archives with Content*/?><?php get_header(); ?><d
iv id="content" class="widecolumn"><?php if (have_posts()) : while (hav
e_posts()) : the_post();?><div class="post"><h2 id="post-<?php the_ID()
; ?>"><?php the_title();?></h2><div class="entrytext"><?php the_content
('<p class="serif">Read the rest of this page &raquo;</p>'); ?></div></
div><?php endwhile; endif; ?><?php edit_post_link('Edit this entry.', '
<p>', '</p>'); ?></div><div id="main"><?php include (TEMPLATEPATH . '/s
earchform.php'); ?><h2>Archives by Month:</h2><ul><?php wp_get_archives
('type=monthly'); ?></ul><h2>Archives by Subject:</h2><ul><?php wp_list
_cats(); ?></ul></div><?php get_footer(); ?>
```

A Page of Posts

A Page Template that displays posts from a specific category depending on what Page is being displayed. This is designed to work with the WordPress Default theme (aka Kubrick), but may work with other themes with a little modification.

Save this to `pageofposts.php`:

```
<?php/*Template Name: PageOfPosts*/get_header(); ?><div id="content" cl
ass="narrowcolumn"><?phpif (is_page('21') ) {$cat = array(12);} elseif
( is_page('16') ) {$cat = array(32);} elseif ( is_page('28') ) {$cat =
array(17);} else {$cat = '';}$showposts = -1; // -1 shows all posts$do_
not_show_stickies = 1; // 0 to show stickies$args=array(   'category__i
n' => $cat,   'showposts' => $showposts,   'caller_get_posts' => $do_no
t_show_stickies,   );$my_query = new WP_Query($args); ?><?php if( $my_q
uery->have_posts() ) : ?><?php while ($my_query->have_posts()) : $my_qu
ery->the_post(); ?><?php//necessary to show the tags global $wp_query;$
wp_query->in_the_loop = true;?><div <?php post_class() ?> id="post-<?ph
p the_ID(); ?>"><h2><a href="<?php the_permalink() ?>" rel="bookmark" t
itle="Permanent Link to <?php the_title_attribute(); ?>"><?php the_titl
e(); ?></a></h2><small><?php the_time('F jS, Y') ?><!-- by <?php the_au
thor() ?> --></small><div class="entry"><?php the_content('Read the res
t of this entry »'); ?></div><p class="postmetadata"><?php the_tags('Ta
gs: ', ', ', '<br />'); ?> Posted in <?php the_category(', ') ?> | <?ph
p edit_post_link('Edit', '', ' | '); ?><?php comments_popup_link('No Co
mments »', '1 Comment »', '% Comments »'); ?></p></div><?php endwhile;
?><div class="navigation"><div class="alignleft"><?php next_posts_link(
'« Older Entries') ?></div><div class="alignright"><?php previous_posts
_link('Newer Entries »') ?></div></div><?php else : ?><h2 class="center
">Not Found</h2><p class="center">Sorry, but you are looking for someth
```

```
ing that isn't here.</p><?php get_search_form(); ?><?php endif; ?></div
><?php get_sidebar(); ?><?php get_footer(); ?>
```

WordPress as a CMS

You can use WordPress for basic content management. If you do, you'll probably create a large number of pages for your content.

Using a Page as the Front Page

WordPress 2.1 introduced the Option to conveniently set any Page as your Front Page. Go to **Settings > Reading** in the Wordpress Admin interface. Under the Front Page Category, you can choose to set any (published) Page or Posts Page as the Front Page. The default setting shows your blog with the latest blog posts.

Alternate Methods for Setting the Front Page for pre-2.1. WP

If you don't want to use the built-in feature to set the home page as a static page, you have two other options. Using the Static Front Page Plugin, it is possible to set any Page as the "front page" of your site. The Plugin modifies the home page query and sticks the Page with a Page slug of "home" to the front page.

When the Page is being displayed as the homepage, if a Page Template with the filename `home.php` exists for your active Theme, the Plugin will override the Page's set Page Template and use `home.php` instead. The Page's set Page Template will still apply if the Page is visited like a standard Page (e.g., `http://example.com/home/`)

As an alternative to the Plugin, WordPress will recognize a "home.php" document in your template directory and use it rather than index.php to theme your front page. However, if your home page isn't your blog, be advised that the home.php template will always apply to the blog page whether you like it or not. Using the query_posts() template tag, you can call on any page before you invoke The Loop. For instance:

```
<?php      query_posts('pagename=home'); ?>
```

Will call up only the post with the pagename "home." See query_posts() for more examples of the template tag in action.

Including a Page

You might also want to include Pages in various places on your site. That way, you can have an easy way to edit elements of your website. There is a Plugin called Improved Include Page that makes doing this easy.

The Dynamic Nature of WordPress "Pages"

A web page can be *static* or *dynamic.* Static pages, such as a regular HTML page that you might create with Dreamweaver, are those which have been created once and do not have to be regenerated every time a person visits it. In contrast, dynamic pages, such as those you create

with WordPress, do need to be regenerated every time they are viewed; code for what to generate has been specified by the author, but not the actual page itself. These use extensive PHP code which is evaluated each time the page is visited, and the content is thus generated on the fly, upon each new visit.

Almost everything in WordPress is generated dynamically, including **Pages**. Everything you and others write in WordPress (Posts, **Pages**, Comments, Blogrolls, Categories, etc.) is stored in your MySQL database. When your site is accessed, that database information is then used by your WordPress Templates from your current Theme to generate the web page being requested. Thus, WordPress information is dynamic, including the information contained in your **Pages**.

An example of a static page might be an HTML document (without any PHP code) you've written as an addition to your dynamically generated WordPress pages, perhaps an "About Me" page. The problem with purely static pages is that they are difficult to maintain. Changes you make to your WordPress settings, Themes and Templates will not be propagated to pages coded only in HTML. The **Page** feature of WordPress was developed, in part, to alleviate this problem. By using **Pages**, users no longer have to update their static pages every time they change the style of their site. Instead, if written properly, their dynamic **Pages** will update themselves along with the rest of your blog.

Despite the dynamic nature of **Pages**, many people refer to them as being static. In the context of web publishing, static and dynamic mean what has been described above. More generally, however, static can mean "characterized by a lack of change". It is easy to see how this definition influenced the word's use in describing types of web pages. It is also easy to see why people think of **Pages** as being static; Posts come and go, but **Pages** are here to stay since **Pages** are typically used to display information about your site which is constant (e.g., information about yourself, description of your site).

In other words, a **Page** contains *static information* but is *generated dynamically*. Thus, either "static" or "dynamic" may be validly used to describe the nature of the WordPress **Page** feature. However, in order to avoid confusion, and because **Pages** *themselves* are dynamic while it is only their *contents* which are in some way static, this document does not refer to **Pages** as being static.

Related content:

- Making Your Blog Appear in a Non-Root Folder

Stepping Into Templates

Template files are the building blocks of your WordPress site. They fit together like the pieces of a puzzle to generate the web pages on your site. Some templates (the header and footer template files for example) are used on all the web pages, while others are used only under specific conditions.

A traditional web page consists of two files:

- The XHTML page to hold the structure and content of the page and
- the CSS Style Sheet which holds the presentation styles of the page.

In WordPress, the (X)HTML structure and the CSS style sheet are present but the *content* is generated "behind the scenes" by various template files. The template files and the style sheet are stored together as a WordPress Theme. To learn more about creating Themes, read Theme Development.

The WordPress Page Structure

A simple WordPress web page structure is made up of three basic building "blocks": a header, the content, and a footer. Each of these blocks is generated by a template file in your current WordPress Theme.

Header

Content

Footer

- The **header** contains all the information that needs to be at the *top* — i.e. inside the `<head>` tag — of your XHTML web page, such as the `<doctype>`, `<meta>` tags and links to style sheets. It also includes the opening `<body>` tag and the visible header of your blog (which typically includes the title of your site, and may also include navigation menus, a logo bar, the description of your site, etc.).
- The **content** block contains the posts and pages of your blog, i.e. the "meat" of your site.
- The **footer** contains the information that goes at the bottom of your page, such as links to other Pages or categories on your site in a navigation menu, copyright and contact information, and other details.

Basic Template Files

To generate such a structure within a WordPress Theme, start with an `index.php` template file in your Theme's directory. This file has two main functions:

- Include or "call" the other template files
- Include the WordPress Loop to gather information from the database (posts, pages, categories, etc.)

For our simple structure, we only need to include two other template files: the **header** and the **footer**. These must be named `header.php` and `footer.php`. The Template Tags that include them look like this:

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

In order to display the posts and pages of your blog (and to customize how they are being displayed), your `index.php` file should run the WordPress Loop between the header and footer calls.

More Complex Page Structures

Header

Content

Sidebar

Footer

Many WordPress themes include one or several **sidebars** that contains navigation features and more information about your website. The sidebar is generated by a template file called `sidebar.php`. It can be included in your `index.php` template file with the following template tag:

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

Where's the Beef?

Notice that we have not included a template tag to "get" *the content* of our web page. That is because the content is generated in the WordPress Loop, inside `index.php`.

Also note that the Theme's style sheet determines the look and placement of the header, footer, sidebar, and content in the user's browser screen. For more information on styling your WordPress Themes and web pages, see Blog Design and Layout.

Template Files Within Template Files

You have seen how WordPress includes standard template files (header, footer, and sidebar) within the `index.php` template file. You can also include other template files within any of your template files.

For example, `sidebar.php` might contain a template file that generates a search form — `searchform.php`. Because this is not one of WordPress's standard template files, the code to include it is a little different:

```
<?php include (TEMPLATEPATH . '/searchform.php'); ?>
```

Instead of using a WordPress template tag to include the file, we'll use the PHP command `include`, which needs to know where the file is located (`TEMPLATEPATH` is a special variable within WordPress that points to the theme's template file directory).

Header

Content

Comment Form

Sidebar

Search Form

Footer

Most WordPress Themes include a variety of template files within other templates to generate the web pages on the site. The following template files are typical for the main template (`index.php`) of a WordPress site:

- `header.php`
    - `theloop.php` (The Content)
    - `wp-comments.php`
- `sidebar.php`
    - `searchform.php`
- `footer.php`

However, this structure can be changed. For instance, you could put the search form in your header. Perhaps your design does not need a footer, so you could leave that template out entirely.

Special Template Files

WordPress features two **core page views** of web pages in a WordPress site. The **single post view** is used when the web pages displays a single post. The **multi-post view** lists multiple posts or post summaries, and applies to category archives, date archives, author archives, and (usually) the "normal" view of your blog's home page. You can use the `index.php` template file to generate all of these types of pages or rely on WordPress' template hierarchy to choose different template files depending on the situation.

The WordPress Template Hierarchy answers the following question:

> *What template file will WordPress use when a certain type of page is displayed?*

WordPress automatically recognizes template files with certain standard names and uses them for certain types of web pages. For example, when a user clicks on the title of a blog post, WordPress knows that they want to view just that article on its own web page. The WordPress template hierarchy will use the `single.php` template file rather than `index.php` to generate

the page — if your Theme has a `single.php` file. Similarly, if the user clicks on a link for a particular category, WordPress will use the `category.php` template if it exists; if it doesn't, it looks for `archive.php`, and if that template is also missing, WordPress will go ahead and use the main `index.php` template. You can even make special template files for specific categories (see Category Templates for more information).

Template File Tips

Here are some tips for creating WordPress template files:

Tracking Opening and Closing Tags

Template files include the use of XHTML tags and CSS references. HTML elements and CSS references can cross template files, beginning in one and ending in another. For example, the `html` and `body` HTML elements typically begin in `header.php` and end in `footer.php`. Most WordPress themes make use of HTML `div` elements, which can also span several files. For instance, the main `div` for the page content might start in `header.php` and end in either `index.php` or `single.php`. Tracking down where an HTML element begins and ends can get complicated if you are developing, designing, or modifying a Theme. Use comments to note in the template files where a large container tag opens and where it closes so you can track which `div` is which at the end of different sections.

Test Template Files Under Different Views

If you have made changes to the comments, sidebar, search form, or any other template file, make sure you test them using different web page views (single post, different types of archives, and pages).

Comment Deviations

If you are designing Themes for public release, keep in mind that someone who downloads your Theme will probably want to modify it slightly for their own use. So, it is helpful if you make notes in your template files where you have made changes from the logic of the Default and/or Classic Themes. It is also a good idea to add comments in your Theme's main style file if you have style information elsewhere (such as in your `header.php` file or in HTML tags).

Close the Tag Door Behind You

If you start a HTML tag or `div` in one template file, make sure you include the closing tag in another template file. The WordPress Forum gets a lot of questions about "what happened to my theme" when they remove the footer template file without closing the tags that began in the header template file. Track down your tags and make sure they are closed. (A good way to verify that this is correct is to test your single and archive page views with an HTML validator).

CSS Styles in Templates

You are free to use whatever HTML and CSS tags and styles you like in your templates. However, you are encouraged to follow the standard WordPress theme structure (see Site Architecture 1.5). This will make your Themes more understandable to your users.

Template File Resources

For a comprehensive list of resources related to Template Files, see Templates. You may also wish to view the other articles in Category:Templates and Category:Template Tags.

Categories: WordPress Lessons | Design and Layout | Templates

Codex Resources

Stepping Into Template Tags

If you take a peek into the `header.php` template file that came with your WordPress Theme, you will notice that where it says "My Blog Name", whatever it is, when you view your WordPress site, it doesn't say "My Blog Name" in the template file. In fact, it has a bunch of strange arrows and parentheses and words that don't make much sense.

This is an example of a Template Tag.

Let's take a few steps toward learning more about what these are and how they work.

What is a Template Tag

A template tag is code that instructs WordPress to "do" or "get" something. In the case of the `header.php` template tag for your WordPress site's name, it looks like this:

```
<h1><?php bloginfo('name'); ?></h1>
```

The template tag is `<?php bloginfo(); ?>` wrapped in an **H1** heading tag. The bloginfo() tag *gets* information from your User Profile and Options> General in the Administration Panels. In the example here, the word `name` inside of the quote marks in the tag instructs the tag to "get the blog's site name". This is called a **parameter**.

Template Tag Parameters

In addition to the *name* parameter in the `<?php bloginfo(); ?>` template tag, there is other information that can be displayed. Let's look at a few of these parameters - and you can find more information and examples on the `bloginfo()` Codex page.

name `<?php bloginfo('name'); ?>`
: As mentioned, this displays the name of the site and is set by the administrator in the Options> General SubPanel by default.

description `<?php bloginfo('description'); ?>`
: This is called the "Tagline" for your blog which is usually some kind of descriptive sentence that says "My blog is about....". It is set by the administrator in the Options> General SubPanel.

url `<?php bloginfo('url'); ?>`
: When you want to display the URL or website address for your WordPress site, you can use URL and it will show up. This also comes from the Options> General SubPanel.

admin_email `<?php bloginfo('admin_email'); ?>`
: If you want to display the email of the administrator, you don't have to type it into the template files. By doing so, it may be open to email harvesters who use sophisticated software to come in and grab email addresses to use for spam. By using `bloginfo('admin_email')`, the email is displayed on the page for the viewers, but the actual email address is disguised from the harvesters. Nice, huh? The administrator's email address is set in the Options> General SubPanel.

version `<?php bloginfo('version'); ?>`
: Sometimes you'd like to show off which version of WordPress you are using. The Themes

that come with WordPress by default include this information in the footer template. It simply displays the version of WordPress your blog uses.

To show the WordPress version, the template tag would look like this:

```
<p>Powered by WordPress version <?php bloginfo('version'); ?></p>
```

Powered By WordPress version 2.8

Notice that only the version number is generated by the *version* parameter, not the words "Powered by WordPress version". Those were written in before the tag so they would be visible on the web page.

To learn more about template tag parameters, see Anatomy of a Template Tag and How to Pass Tag Parameters.

How Do You Use Template Tags?

Going through the various template tags in the Template Tags menu on the Codex, you will see that many of them are very simple, like the `bloginfo()` template tag, but many look very complicated to use. Let's look at some examples of how they are used to help you understand the "language" of the template tag codes.

As we saw in the `bloginfo()` template tag, all it took was one word to change the output of the tag. This word is called a *parameter* and it instructs the template tag to *do* or *get* something. In this case, the instruction is to *get name* which displays the site's name.

The template tag `the_title()` displays the title of the post, usually at the top of your post article. This tag *gets* the post title and displays it, by default, but it also has a *do* in the parameters which will help you change the look and presentation of the post title.

By default, the tag looks like this:

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

And the results look something like this:

Using WordPress Makes Me Smile

Let's say you want to put some kind of reference that highlights the title in some way, like a graphic or character entity like an arrow or bullet. Let's put a yen sign, ¥ ,the sign for Japanese money, in front of our title.

If you look carefully at the instructions for the tag `the_title()`, you will see that the parameters are:

```
<?php the_title('before', 'after', display); ?>
```

We want the yen sign to be *before* the title, with a space after the yen sign and before the title, so let's add it to the parameters:

```
<?php the_title('&yen; '); ?>
```

Which, when the page is generated, would look like this:

¥ Using WordPress Makes Me Smile

Now, let's take this a little further and put something after the post title. Let's say you want to encourage people to read so we'll add a little incentive arrow (») to motivate them.

```
<?php the_title('&yen; ', ' &raquo;'); ?>
```

Notice, we added another space before the arrow to separate it from the post title when the page is generated for viewing.

¥ Using WordPress Makes Me Smile »

You can also style your title heading in many different ways. Here is another example using heading tags.

```
<h2><?php the_title('Post Title: '); ?></h2>
```

We've put the entire post title into an H2 heading and added the phrase "Post Title" to the beginning of the post title.

Post Title: Using WordPress Makes Me Smile

Note: Not all template tags take before and after arguments, though the_title does. Check the codex page for the specific tag you're using to see what arguments it accepts.

Boolean Template Tags

The above template tag example uses simple parameters separated from each other with quote marks and commas. Now consider examples of Boolean Template Tags that connect more than one parameter together using boolean math techniques. One common boolean expression uses the "and (&)" logic to connect the parameters.

The template tag wp_list_cats() is commonly found in the WordPress sidebar or menu template file. It lists the site's Categories.

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

By default, some of the template tags' parameters are:

- *all* - Displays all of the Categories
- *sort_column* - Sorts by Category ID
- *sort_order* - Sorts in ascending order
- *list* - Sets the Categories in an unordered list (`<ul><li>`)
- *optioncount* - Does not display the count of posts within each Category
- *hide_empty* - Based upon the first two parameters (optionall and all), does not display Categories without posts
- *use_desc_for_title* - Uses the Category description as the link title
- *children* - Shows the children (sub-Categories) of every Category listed

An example of this category list might be:

- Stories About My Life
- Stories About My Family
- Things I Want To Share
    - About WordPress
    - About Writing
    - About Story Telling
- Facts and Fiction About Life

The indented list with "About WordPress", "About Writing", and "About Story Telling" are the **children** or sub-Categories of the **parent** Category "Things I Want To Share". These titles, by default, are not the actual titles of the Categories, they are the **descriptions** of the Category you set in the Administration> Manage> Categories panel.

If you would like to show the actual title of the Category, instead of the Category description, change the template tag to:

```
<?php wp_list_cats('use_desc_for_title=0'); ?>
```

The zero sets the parameter to **false**, turning off the use of the description as the title. Now the Category titles would appear:

- My Life Stories
- My Family
- Sharing
    - WordPress
    - Writing
    - Story Telling
- Facts and Fiction

Let's say that you don't want the sub-Categories for "Sharing" to appear on your list. You would then add the parameter to not show the children, along with the parameter for showing only titles and not descriptions, with the boolean "and" using the ampersand (`&`).

```
<?php wp_list_cats('use_desc_for_title=0&children=0'); ?>
```

Notice there are no spaces around the ampersand. All the parameters run together without any spaces or quote marks in between, just around the whole parameter. Now the Category titles would appear as:

- My Life Stories
- My Family
- Sharing
- Facts and Fiction

As another example, if you want to display the Category links as the Category title, sort the list alphabetically by name, show the number of posts within each Category, and only show the *children* (sub-Categories) of Category ID number 3 ("Sharing"), the template tag would look like this:

```
<?php wp_list_cats('sort_column=name&sort_order=asc&optioncount=1&use_desc_for_title=0&child_of=3'); ?>
```

- Story Telling (21)
- WordPress (23)
- Writing (10)

Template Tags and The Loop

Many of WordPress' template tags work within the WordPress Loop. This means that they are included in the template files as part of the php "loop" that generates the pages the viewer sees based upon the instructions inside of the Loop.

The WordPress Loop begins with:

```
<?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>
```

Template tags that work within the loop must be in the middle area here, before the ending section of the Loop below:

```
<?php endwhile; else: ?><p><?php _e('Sorry, no posts matched your crite
ria.'); ?></p><?php endif; ?>
```

Template tags that need to be inside of the loop include `the_content()`, `the_excerpt()`, `next_post()`, and `previous_post()`. If the template tag you want to use doesn't have to be within the Loop, like `wp_list_cats()` and `wp_list_pages()`, then you can put it anywhere you like, for instance in the sidebar, header, or footer template files.

Learning More About Template Tags

This is just a tiny step into learning about the various powerful template tags WordPress uses to generate your website. You can learn more about the different template tags WordPress uses in the following articles and resources.

- WordPress Template Tags Catalog
- Templates
- **Stepping Into Template Tags**
- Anatomy of a Template Tag
- How to Pass Tag Parameters
- The Loop
- Include Tags
- Conditional Tags

Styling Your Template Tags

- Styling Lists with CSS
- Lessons: Next and Previous Links
- Lessons: Separating Categories
- Lessons: Styling Page-Links
- Lessons: Good Navigation Links
- Formatting Date and Time

External Resources

- fr:Decouvrir_les_Marqueurs_de_Modele

Using Images

A picture says more than a thousand words.
WordPress makes it easy for you to add images to your WordPress site. You can upload them directly from within WordPress by using the built-in file uploading utility in the post screen. Or you could use any FTP Client software to upload many images to your WordPress site.

The Quicktag buttons feature an **image** link, making it easy to link to images from within your post as you write it. If you used the inline upload feature, your picture will be in the /wp-content/uploads folder, unless you've specified another folder on the Miscellaneous admin panel.

WordPress can now (within the posting page) resize images and create thumbnails. There are also photo galleries that can show many images without adding each one separately to a page.

And if you choose to let the images speak for you, consider creating a PhotoBlog or Gallery.

Styling Images in WordPress

Current versions of WordPress now have image alignment built-in. WordPress adds CSS classes to align the image to the right, left, and center of a paragraph, so the text will wrap around the image.

In order to take advantage of these new CSS classes for image alignment and the text wrapping around the image, the WordPress Theme must include the following in the `style.css` found in the WordPress Theme directory.

```
img.alignright {float:right; margin:0 0 1em 1em}img.alignleft {float:le
ft; margin:0 1em 1em 0}img.aligncenter {display: block; margin-left: au
to; margin-right: auto}a img.alignright {float:right; margin:0 0 1em 1e
m}a img.alignleft {float:left; margin:0 1em 1em 0}a img.aligncenter {di
splay: block; margin-left: auto; margin-right: auto}
```

When adding the image in your WordPress blog, select the image alignment as right, left, or center in the Image/Media Panel.

The image will be embedded into your blog post with the selected style for alignment such as:

```
<img src="http://example.com/images/leaf.jpg"  alt="leaf graphic"  titl
e="leaf graphic" class="alignright size-medium wp-image-3109" height="2
5" width="30"/>
```

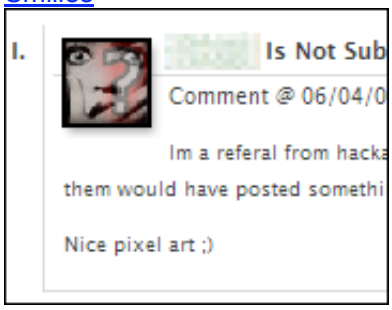
For more information on styling images in WordPress, see Wrapping Text Around Images.

Images Resources for WordPress

File:imagesinarticlesidebar.jpg
Example of images in background, sidebar, blockquote, and post
There are a variety of WordPress Plugins which add image functionality, utilities, and galleries to

your website. Some run from within WordPress while others stand alongside.

You can also add images to your WordPress site through the use of template tags, Plugins, and in the style sheet of your Theme. For example, you can add images to your:

- Categories
- Comments with Gravatars
- Graphics Symbols
- Header
- Headings
- Smilies
- 

  Gravatar used in Comments
- Sidebar
- Footer
- Lists
- Menus
- Between posts
- Between lists
- Graphics in Feeds
- Links and Blogrolls
- Next and Previous Links
- Template Tags - Many feature image parameters

Using Images in Posts

Images can be used in a variety of methods in your WordPress posts and Pages. They can be a major subject, or a referenced detail that enhances the information or story.

The first thing you should consider is the "look" of the images on your page. Not what the images are of, but the general look of how they flow and interact with the rest of the content on your page. Wrapping Text Around Images helps you to begin to understand how images interact with the text around them, changing the margins, padding and borders around the images within the content. It will also help you understand how to create captions under your images.

The next thing to consider is the size of the images. There are two ways of actually **sizing** an image. It is either the size that it is, or a thumbnail link which, when clicked, takes the user to a new page with an enlarged image of the graphic.

Inserting Images into Posts

Inserting an image into a post still seems to confuse a lot of people.

There are 2 steps involved to inserting an image into a post. First, the image file must be uploaded onto your web server before it can be inserted into a post. The second step is to actually insert the image into the post in the appropriate location.

The simplest way to do this is to use the "Add Media" function on the post screen (beside the "Visual" and "HTML" tabs). Choose the appropriate button depending on whether you are adding photos, videos, audio, or miscellaneous media (e.g. PDF files). This method will complete both steps as outlined above.

In the "From Computer" area of the screen you need to upload the desired image file. NOTE: Before hitting the "Upload from Computer" button it is recommended that you select the "Browser Uploader" instead of using the default Flash Uploader. After selecting the "Browser Uploader", hit the "Browse..." button and choose the desired image file on your computer. Then, hit the "Upload" button. Your image file is now on your web server.

Make sure you give the image an appropriate title, as well as a relevant description if desired. Choose the desired alignment and size, and hit the "insert into post" button. Your image is now inserted into the post at the location where your cursor was last active.

Every time you upload an image to your web server it is added to the "Gallery" of images that are available for that blog. If you want to insert the same image into another post go to the "Gallery", hit the "Show" link, and then follow the same instructions to insert that image into another location or post.

You can also manually upload an image onto your webserver with an ftp program. The details for this method are not included here.

Image Size and Quality

The size and quality of an image for use on a web page is determined by a variety of things.

Physical Size
: The physical size of an image is based upon two things: The size of the image on the screen and the file size. Generally, the file size is treated as a different issue.

File Size
: This is the size of the file on your hard drive or server.

Resolution
: *Resolution* refers to the number of pixels in an image. Resolution is sometimes identified by the width and height of the image as well as the total number of pixels in the image.

File Type
: There are basically X image types popularly found on the Internet: `jpeg`,`gif`,`png` and (for favicons (the icons next to the address)) `ico`.

The **physical size** of the image is information we need to know in order to determine how much "space" will the image occupy on a web page. If your WordPress Theme features a fixed width content area of 600 pixels and the image you want to use is 800, the image will push the sidebar and layout of your web page around, messing up your design. Images within that 600 pixel width need to be restricted to that maximum width in order to protect the layout of your page. It's up to you to determine what size they should be from there, matching the image to

your overall layout and styles.

File size dictates the time it takes to load your page, the larger the file size, often increased because of a high **image resolution** quality, the longer it will take to load. People often don't have the patience to wait through long web page loads, so keeping your file sizes low speeds up your web page access times. Typically, large high quality images should be kept between 100K and 60K. Smaller images should be closer to 30K and lower.

The **resolution** of the image dictates its clarity. The higher the resolution, though, the larger the file size, so you have to make a compromise between quality and file size.

Luckily, the various file types most commonly used on the Internet have *compression* features. When you save the file as one of these types, it condenses or *compresses* the data information in the image file. Internet browsers can *decompress* this information to display the image on the screen. Some graphic software programs allow you to set the compression rate to control the quality of the image (and file size) at the time you save it. Depending upon your use of the images on your site, you may have to experiment with this to get the right ratio that keeps the resolution quality good while maintaining a small file size.

Websites use four common **file types**. The end of a filename (called the *extension*) tells what type it is. One type, `ico`, is to make a *favicon* file -- but this is usually only done when a website is first set up. The other three types are used for general images:

- `jpg` (JPEG) is good for photographs. Saving a photo as `jpg` removes detail from the photo. Good photo editors let you control how much detail is removed (the "compression"). Different photos need different compression; doing this carefully and viewing the result can give you a usable photo with a small file size.
- `gif` can be poor for photographs. It's better for line art, like logos, with solid areas of the same color.
- `png` is for both photographs and line art. It compresses photos without losing detail, but usually makes larger photo files than JPEGs. Some older browsers don't completely support `png`, though.

If you aren't sure which file type is best for a particular image, try saving the image in more than one type and comparing the file sizes. Using the right type can make a big difference! There's more information in Sitepoint's GIF-JPG-PNG What's the Difference article.

Resizing Images

Not all graphic software packages allow you to resize images, though most should. Check your graphics software table of contents or index for *resize*, *size*, *transform*, *reduce*, or *enlarge*, all synonyms for the for the same thing. If they don't have the feature, you may have to find different software.

The process of resizing images is fairly simple. There are usually two methods:

1) You can resize an image through the use of tools provided which allow you to manually shift the edges of an image to deform or resize the image. The best way is to grab a corner, not the edge, to resize the image. The corner "handle" will usually resize the image maintaining the overall height-width ratio. Check your manual for specific instructions.

2) The other method involves simply specifying the image's final size. The advanced graphics

programs allow you to set it by exact dimensions or a percentage of reduction or enlargement.

After resizing the image, the image may be smaller, but it may also be slightly out of focus. You can sharpen the focus of the small image by using the **sharpen** feature in your software.

When you have fine-tuned your small sized image or new thumbnail, export the image as a `jpg, gif,` or `png`.

Styling Images

Images can have borders, frames, captions, and be styled in many different ways. There are basically two ways to style an image on your site. You can style it from within the style sheet or *inline* on a specific image.

Styling All Images

Styling your images from within the `style.css` of your WordPress Theme can cover the styling for every image on your site, or specific images.

To style every image on your site to look a particular way, look for or add the CSS selector for the `image` tag. Then add your styles to the tag. For instance, let's say that you want a black border around all of your images and you want space between the border and the image, as well as the appropriate spacing around the image and the text.

```
img {     margin: 5px;     padding: 10px;     border: solid black 1px}
```

Maybe you want something a little more dramatic. You can change the border thickness and set it to a "double" line. And maybe you really want to isolate your image from the rest of the text, so you increase the margin around the image.

```
img {     margin: 20px;     padding: 10px;     border: double black 1px
}
```

Styling Some Images

You can add to your style sheet a specific style for certain images. If you have already styled all of your images, you must make sure you specify every style declaration or attribute specified in the `image` tag style in order to override that attribute. If you do not change the margin, then it will remain the same in the new style. This is called the CSS Parent/Child Relationship.

Let's say you would like to have some images in your posts filed in the category of Nature have a nice green background. The rest of the images should look the same, just the ones you add in your Nature category. You simply add a `class` to your style sheet and each image within that category.

To make it easy to remember, we'll call our `class`"nature". We want to have a very dark green thick border and a medium green background around the image to highlight it.

```
img.nature {     margin: 20px;     padding: 20px;     border:solid #003
300 4px;     background: #006600;}
```

On each of the images within that category, you simply add the `class` for "nature":

```
<img src="leaf.gif" alt="Red leaf" class="nature" />
```

If you need more styles for different images, you can create more of them as needed.

Styling One or Two Images Inline

There are times when you just want one or two images on your site to look different from the rest. This technique is called **inline styles.** It applies the CSS styles directly to the image itself.

For example, say you want to have an image isolated against a back background to call attention to it.

```
<img src="leaf.gif" alt="Red leaf" style="padding:20px; background: bla
ck; border: none" />
```

This is not a technique to use on every image. It is to be used on occasional images that need a "little something special."

Resources

- Wrapping Text Around Images
- WordPress Design and Layout
- Photoblogs and Galleries
- Image File Formats on the Web

Finding WordPress Help

Besides the helpful WordPress forums and this Codex, there are many sites dedicated to helping WordPress users use WordPress even better. WordPress help is everywhere. So how do you find it when you really need it?

The WordPress FAQ provides extensive answers to frequently asked questions. We've even included a document on how to use the WordPress support forums to get better results on your requests for help.

Before posting in the forums, since it is run by volunteers working hard over long hours when they could be doing something much more constructive, maybe you should start with a **search**.

To increase your search capabilities, you can add the Codex Searcher Plugin and search the Codex from your WordPress Administration Panels. Click on one of the search results and the page will open in a new window or tab so you can have the article open while working on WordPress.

Searching The Net For WordPress Help

There are a variety of ways to search for the information you need. The biggest problem is finding the "words" that describe your problem. The next biggest problem is limiting your search to only WordPress resources or sites. Let's look at how to do this.

Search Using Keywords

Sometimes it's easy to search using keywords because the problem you are having tells you what is wrong with it. Copy the most important words from the error, paste them into a text editor and take a good look at them. Here is an example:

```
Warning: main(/home/atlantis/public_html/wp-includes/functions.php): fa
iled to open stream: No suchfile or directory in /home/stargateatlantis
/public_html/wp-settings.php on line 67Fatal error: main(): Failed open
ing required '/home/atlantis/public_html/wp-includes/functions.php' (in
clude_path='.:/usr/lib/php:/usr/local/lib/php') in /home/stargateatlant
is/public_html/wp-settings.php on line 67
```

Hidden within this information are the key words you need to help you get the answers.

Opening the files referenced, like `functions.php`, might help, but other files are often processed long before it gets to `functions.php` or `wp-settings.php` and the line number related to the generated file is not necessarily the line number in those template files. It might get you close, but maybe not.

From this error, though, you have some critical information to help you with your search. You know the names of the files that are causing the grief:

- functions.php
- wp-settings.php

These are part of the *keywords* that you will need to be looking for.

The specific errors are **failed to open stream** and **failed opening required** and they both happen within the `wp-settings.php` file. Maybe the problem isn't in the `functions.php` but

in the `wp-settings.php` file? And you see the words "failed" and "open" in both errors. That's a clue.

Create a search in your favorite search engine that included the words:

```
wordpress failed open wp-settings.php
```

This should get you started on narrowing down the problem.

But not all keywords can be so easily found. If the problem is with CSS or HTML, you can include the specific tag or selector that has the problem, but this might not be the problem at all, so you need to hunt for words that describe your problem.

If there is a difference in your web page layout in Microsoft Internet Explorer and another browser, then search for information on **Internet Explorer layout bugs**. If you have a consistent error in the different browsers in the layout, which part of the layout is it affecting? The *sidebar*, *header*, post *content*, or *comments*? If the problem is the image in the header not showing up or not working right, then search for **wordpress header image** to start, then add the specifics like **wordpress header image missing** to narrow things down.

Brainstorming Search Terms

When really struck for keywords, you can try *brainstorming* your keywords. First, write down the problem. Be as descriptive as you want:

> *I'm having trouble with the nested list in the sidebar of my layout. It isn't lining up the items under the titles right. It is keeping things on the left margin when I want them to be indented.*

Right there, you've listed keywords to search with within your description. There are:

- sidebar
- layout
- nested list
- left margin
- indented

Arrange those in different ways and you have some keywords to search with.

A second method is to talk it out with someone. It doesn't matter of the person you talk to is a WordPress expert or not, just talking to them will force you to use very simple terms and references, and *among those are your keywords*. Your friend might respond to your description with:

> *"Oh, you are trying to change X to B and get the result P. Why didn't you say so?"*

There, you have your search keywords.

Searching Tips and Tricks

Once you have some starting keywords, it's time to put them to work. Remember, you are not stuck with your starting keywords. They are just that, a start. As you dig into the information, you may replace those words with ones that narrow the field down a little. For example, if you are looking for "wordpress sidebar layout nested links", you may discover that the problem lies within the specific Theme you are using, add the name of the theme to your keywords and it may narrow down your search.

Let's look at some more tips for improving your odds of finding a solution.

Search Engine Site Search

Did you know you can search a specific site from a search engine?

Google Site Search

To search a specific website, like **wordpress.org**, in the Google search window, type your keyword and end with the *site:* reference:

```
keyword1 keyword2 keyword3 site:wordpress.org
```

Google will search all **wordpress.org** sites such as *www.wordpress.org*, *wiki.wordpress.org*, and *codex.wordpress.org*.

To narrow down your search to a specific site, like *www.wordpress.org'*, use the *site:* reference and type:

```
keyword keyword keyword site:www.wordpress.org
```

Google will search only the www.wordpress.org site, which includes the Support Forums, but not the WordPress Codex or Wiki.

Yahoo Site Search

To search a domain or website specifically in Yahoo, their Yahoo's Advanced Search Page allows searching by domain name directly.

Other Search Engines

Other search engines provide a variety of ways to narrow your search to a specific site. Check with their **Advanced Search Options**, if they provide them, or at the Search Engine Watch's Search Command list for details on searching by domain or url.

Use Quotes to Group Keywords

You can group different keywords together to narrow your search. For example, instead of looking for:

```
sidebar layout nested list left margin indented
```

You can group key phrases together with quote marks:

```
sidebar layout "nested list""left margin" indented
```

This would limit your search to anything with the words *sidebar*, *layout*, and *indented*, and the phrases "*nested list*" and "*left margin*".

You can also use Boolean references and a technique with plus and minus symbols to group keywords together. There are a lot of options for improving your searching techniques and you can learn more about these:

- Search Engine Watch's Web Searching Tips
- Internet Tips: Searching the Internet
- Tutorial – Guide to Effective Searching of the Internet

- Windweaver's Seach Yahoo Tips

WordPress Sources for Help

Of course, your best chance of finding WordPress information is to go to the source. The following are the main places to go to get WordPress help and support:

- WordPress Codex - WordPress Online Manual
- WordPress Support Forums
- IRC Freenode WordPress Support on channel #wordpress
- WordPress IRC Live Help

Other Helpful Resources

- Have you tried your favourite search engine yet?
- Search the support forums. There are some WP-geniuses over there that will try to help you out. Please read Using the Support Forums and **Finding WordPress Help** to find out how to search the forums and post a clear, answerable question.
- If you prefer visual learning, try these free WordPress video tutorials (requires email registration) provided by EasyWebTutorials.com
- PHPbuilder has how-to articles, a library of PHP code snippets, and a discussion forum.
- Refer to the external FAQ site : The fledgling FAQ has quite a few answers for the troubled.
- Check out Podz' tutorial collection : Podz is our lead support moderator, and Codex contributor. He might be working on the help you are seeking right now.
- Using the Live WordPress Support on the IRC. See: WordPress IRC Live Help and IRC.
- Search the Mailing Lists and their archives. If you need professional support, you may try mailing the Wp-Pro mailing lists, and invite bids for your project.
- If all else fails, go to the WordPress Trac (bug tracker) and see if your problem has already been addressed by searching the bug database. If you think your problem merits a new bug report, file one.

I Make Changes and Nothing Happens

Things are going great. You have figured out how to write a post, how to make a few categories, and maybe even add a plugin or two. Then **it happens**. You make a few changes in your WordPress Theme and when you view your WordPress site, *nothing has changed*. Your fix isn't fixed. Your change isn't changed. NOTHING HAS CHANGED! You want to scream, pull out your hair, pound on your computer, and shout blame.

Hang on. **Calm down.** A great many errors are in fact mistakes born of haste. Even if you're operating under a tight deadline, try to **remain calm, and collected**, as you proceed. It will help. If you start to get unduly tense, or frustrated, simply get up and walk away for a few minutes. Sometimes all it takes is a fresh, rested pair of eyes to spot a missing semi-colon, or an extra quote mark, or to realize that the solution is much simpler than you originally thought.

The problem might be with WordPress, it might be with your database, it might be with your server, or it might be something you did to screw things up, but the reality is that a lot of the time it is none of those things. The culprit is probably your Internet browser.

Whatever the problem is, we're here to help. Let's look at the possible problems and solutions for what to do when you make a change and nothing happens.

The Browser Cache

Did you know that a computer is supposed to make your life easier? Less complicated? It is supposed to save you time and energy and actually improve your life. No? Well, maybe not, but your Internet browser does its best to try to make your life a little easier.

When you first visit a web page, it often takes a while to load, right? But the next page you visit within that site doesn't take so long to load. This is because, in an effort to be helpful, the browser stores the information on your computer so it reloads it from your computer, not from the actual site. This is called the **cache** and it is meant to speed up your Internet browsing.

The term *cache* may sound familiar. Remember the pirates and thieves of old who would stockpile their treasures in a cave, hole in the ground, or somewhere "safe". Called the *cache*, the Internet browser stores files and information for the browser to reuse when the page is refreshed or viewed again.

The problem comes when you make a small change to your site and the browser doesn't recognize it as a significant change, so it reloads the same page you just looked at. The solution is to clear or empty your **browser's cache**.

Clearing the Browser Cache

Normally, to see the changes on your page, you click the **Refresh** button on the browser toolbar or press the F5 key on your keyboard. In many cases, this simply reloads the page without clearing the browser's cache. Here are some techniques to wipe clean the browser's cache so you will see the changes when your page reloads.

Microsoft Internet Explorer

1. Hold down SHIFT and click on the REFRESH button in the toolbar under the menu.
2. **For Serious Clearing:** If you are having problems clearing out the cache, then force it by choosing from the menu TOOLS > INTERNET OPTIONS > TEMPORARY FILES. Click on **Delete Temporary Files**. You can choose the checkbox to delete **all** Internet files, but you might not want to as that will also clear all your passwords and cookies, but if you are having trouble viewing the changes on your page, go all the way.

Mozilla Firefox

1. Hold down CTRL+SHIFT+R.
2. If you are using Chris Pederick's Web Developer Extension, click Miscellaneous and then Clear Cache.
3. **For Serious Clearing:** From the browser menu, Tools > Options > Privacy > Cache and select Clear.

Netscape
: From the menu, click Edit > Preferences > Advanced. Choose "Cache" and click both "Clear Memory Cache" and "Clear Disk Cache".

Mozilla 1.x and up
: From the browser menu, Edit > Preferences > Advanced and click "Cache" and "Clear Cache".

Opera
: From the browser menu, Edit > File > Preferences > History and Cache and click "Cache".

Safari
: From the browser menu, Safari > Reset Safari and click Reset to confirm OR Safari > Empty Cache.

Miscellaneous
: Each browser may have a way of stopping or minimizing the caching of web pages. Using this technique will definitely slow down your web page viewing, and it isn't a perfect solution because some caching may still occur, but it does help. Check your Internet browser's help files for the specifics on how to turn off the cache feature.

A WordPress Cache Plugin

Some WordPress plugins also add cache functionality to your blog. This helps your blog load faster because WordPress can retrieve the pages of your blog from the cache instead of generating them all over again.

All good cache plugins will clear the cache when a post, page, or comment is published. However, if you make other changes (e.g. to your theme), the cache may not be cleared and the old version may still appear. In this case, check the plugin's instructions to find out how to clear its cache.

Note that WordPress does *not* come with a cache by default, so the above would only apply if you installed a cache plugin yourself.

Check Your Source

You know, even the very best web page designers, developers, and programmers screw up. It's the little details, the forgotten semi-colon, the misspelled tag, the lack of attention to a detail that screws things up. If the best do it, then it's very possible you have overlooked a little detail. And if you did, well, welcome to the club. It's a part of the process. Let's look at some of the most commonly overlooked details that happen when you aren't paying attention.

Check the Address

Is the name and folder for the file you "fixed" the same as the one you are viewing? Look at the following two addresses (URLs).

- `wordpress/wp-content/themes/yourtheme/style.css`
- `test/wordpress/wp-content/themes/yourtheme/style.css`

In this case, you can probably see the difference, but when viewed in an address bar or in a text editor, you might miss the word `test` that sets the folder.

Pay very close attention to the difference between `style1.css` and `stylel.css` if you are using different style names, too. The first filename is `style` followed by the digit one, while the second filename is `style` followed by a lowercase L. If you are working with different but similar files, make sure you give them distinctive names like `style-red.css` and `style-800.css` so you can clearly see the difference.

Check the Template

If you're editing a template, are you sure the page you're viewing is being generated from that template? Remember that many templates have very similar text on them; for example, a post header may appear on a single post page, index page, search page, archive page, and others.

See Template Hierarchy if you're having trouble figuring out which template is in use.

Check Your Upload

When you make a change in a file, it is often on your computer's hard drive and you have to upload the file to your host server in order to view it on the Internet. Did you actually upload it? Did you put it in the right folder? Is it really there? When over-writing the exact same file, it doesn't always do a complete over-write, so consider deleting the original on the host server and then uploading the new version to make sure the right and whole thing is there.

Test Yourself

If you still can't see the changes you made, and the file is in the right place with the right name, and you are sure it's the right file, then go through these steps:

1. Make a backup of the file you are working on and check that the backup is in a safe

place.
2. Make a big change (such as setting the background in your `style.css` as `#ff0000` or even red).
3. View the changed web page in your browser. Make sure you clear the cache to be sure you have the new version.
4. If nothing changes, delete the file (and only that file) from the server and try to view the file again. If nothing continues to change, you and WordPress are looking at completely different files. It's time to get out your detective hat and start tracking down what is going on and where your files went to.
5. Check your URL settings in your Options Panel and also in the database, and if this continues to be unsolvable, post a note explaining what you've done and what's the result on the WordPress Forum and let the experts step in to help.

Debug It

Programmers use the term **debug** to describe the process of going through code and finding the little criminal that is messing things up. We're going to look at how to debug your CSS, HTML, and PHP code to help you figure out why you can't see the changes you have made.

When debugging a problem, **change only one thing at a time**. If you're not sure if the problem is in line 37 or line 40, don't change both lines in one go! First change line 37, and see if that fixed the problem. If not, *undo* the change you made on line 37 and then make a change on line 40. It's important to *undo* proposed fixes before trying something else, even if the first attempt doesn't immediately introduce new problems.

Every time you make a change to a file, you run the risk of accidentally making more mistakes. These sorts of things tend to cascade, making debugging an infuriating process. Do one thing at a time.

Debugging CSS

Debugging your style sheet, or your CSS, can be challenging because you have to find the area in the HTML that is causing the trouble and then track that section back to its style in your `style.css` file. The article on Finding Your CSS Styles will help you find those troublesome sections.

Once you have found the style that is causing the problem, you need to figure out what it is about the style that is causing the problem. Here is a quick checklist of things to consider as you troubleshoot your CSS:

- Is everything spelled correctly?
- Is every period, brace, colon and semi-colon in its right place?
- Are you using the style attribute or *declaration* correctly?
- Is there a declaration that shouldn't be in there, like `font-weight:x-large`? The declaration `x-large` is used for `font-size` not `font-weight`.
- Do you have spaces in places where they shouldn't be, like inside of a `background-image:url ( ' bg.gif ' )`? (correct: `background-image:url('bg.gif')`)

You can find more information to help you debug and troubleshoot your CSS with:

- CSS Troubleshooting
- Finding Your CSS Styles
- CSS Fixing Browser Bugs
- Troubleshooting Your Theme - *Coming soon*

Debugging HTML

Similar to debugging CSS, HTML can also get bogged down with careless little mistakes like misspellings, forgotten closing tags, forgotten < arrows, and other little errors that can send your site into twisted remains.

It is highly recommended that you use one of the HTML Validators available for free on the Internet. Also, Mozilla Firefox has a powerful free add-on (actually bundled with most Firefox installations) called the Web Developers DOM Inspector that will help you validate and fix your website problems very quickly and easily.

Some tips for debugging your HTML/XHTML may include:

- Improperly nested XHTML, especially in nested lists commonly found in the sidebar.
- Unclosed tags.
- Self-closing tags not closed by use of the forward slash (example: `<img src="ball.gif" />`).
- Incorrect tag usage.

For more help on debugging your HTML, check out these articles and resources:

- Validating a Website
- Stepping Into Template Tags

Debugging PHP

If you have access to your webserver's error log, take time to check it. PHP usually logs informative errors here when things go wrong. These log messages can sometimes be a little cryptic, but they should always give the line number of the offending piece of code.

Unfortunately, what PHP thinks is the offending piece of code is sometimes not the problem. For example, an unclosed curly brace `{` may be reported as a problem on some line number way on down in your script. A quick cheat sheet of PHP error messages and their common causes can be found at Common PHP Errors.

To debug your PHP, here are the steps you should follow:

See Where You Are

The `die()` command is probably the single most useful debugging tool available. `die()` immediately halts execution of the program, optionally displaying a message of your choosing.

When trying to work through problems, sprinkle `die()` statements in at key sections of your script, giving each an informative message:

```
die('Stage One Complete');
die('Disinfribullation Complete');
die('Finished Collating');
```

Load your program, and see which message (if any) gets displayed. If you see the first message, you know your program got that far without error. You can safely remove (or comment out) that `die()` command, and re-run your program, to see how much farther it got. Step through your program this way until you've isolated the problem area.

This technique has some limitations, though. First, if **nothing** is being displayed in your browser, then most likely you have a fatal syntax error somewhere in your script. Check your webserver's error log, if possible. If the script executes completely -- but incorrectly -- and none of your `die()` messages are displayed, then you've got some more work to do.

See What's What

If things aren't being set as you want them, or stuff isn't happening when you want, you'll likely need to check the value of different variables at different places of your script. Simply pass the variable to a `die()` command, and you'll be able to see what that variable's value is:

```
die($user_level);
```

This will display the value of the variable $user_level, which should reveal the current user's user level.

This technique is fine for some variable types, like boolean and string ("scalar") variables in programmer parlance; but fails for arrays and objects. To see the value(s) of arrays and objects, use the print_r command:

```
print_r($post);
```

It will display the value of the $post array in a human-readable format:

```
Array ( [0] => stdClass Object ( [ID] => 1 [post_author]
=> 1 [post_date] => 2005-02-16 09:16:46 [post_date_gmt]
=> 2005-02-16 14:16:46 [post_content] => Welcome to
WordPress. This is your first post. Edit or delete it,
then start blogging! [post_title] => Hello world! [post_category]
=> 0 [post_excerpt] => [post_status] => publish [comment_status]
=> open [ping_status] => open [post_password] => [post_name]
=> hello-world [to_ping] => [pinged] => [post_modified] => 2005-04-15
08:45:42 [post_modified_gmt] => 2005-04-15 13:45:42
[post_content_filtered] => [post_parent] => 0 [guid]
=> /?p=1 [menu_order] => 0 ) )
```

This format allows you to see what each key/value pair is inside the array.

Using `print_r()` does not stop execution of the program, so it is often necessary to call `die()` immediately afterwards.

Use `print_r()` and `die()` to check the values of your variables through the execution of your script. There's also a `var_dump()` function which works similarly. Use whichever mechanism you find most informative.

When All Else Fails

If all else fails, know this: **You are not alone.** We've all been there. To help you get back on track and your site up and running again, check the following resources for more help:

- Troubleshooting
- WordPress Lessons

This article is marked as in need of editing. You can help Codex by **editing it**.