

# 13

## Administrator's Reference

This chapter will provide information to help you with the WordPress administrative tasks. A few topics that have been covered elsewhere in the book are explained in greater detail here.


I'll review the essentials, and then give you some important links that you can visit for more details. This chapter is a kind of "cheat sheet" to which you can refer for quick answers to common administrative issues.

### System requirements

The minimum system requirement for WordPress is a web server with the following software installed:

- PHP version 5.2.4 or later
- MySQL version 5.0 or later

Although Apache and Nginx are highly recommended by the WordPress developers, any server running PHP and MySQL will do.

[  Learn more about system requirements at <http://wordpress.org/about/requirements/>. ]

Your system will also need to be set up in a particular way if you want to use pretty permalinks, which give you nice-looking URLs throughout your site.

## Enabling permalinks

We talked about permalinks in *Chapter 2, Getting Started*, so please start by giving it a quick glance to refresh your memory.

Permalinks are a great functionality in WordPress. They allow you to take advantage of great-looking URLs, but in order to use them, you must have an Apache web server with `mod_rewrite` enabled.

Most commercial web hosts have this module turned on by default, so there's no need to do any additional tuning up. However, for a local installation, if you are using Windows IIS, for example, then some tweaks might be required. There are ways to implement permalinks on IIS; the most straightforward being to buy Helicon's ISAPI\_Rewrite ([http://www.helicontech.com/isapi\\_rewrite/](http://www.helicontech.com/isapi_rewrite/)). There are other ways as well; start at the following codex link. This topic is too extensive to be explored in depth in this book, but I encourage you to search the Internet for other people's solutions.



Learn more about permalinks at [http://codex.wordpress.org/Using\\_Permalinks](http://codex.wordpress.org/Using_Permalinks).

## The importance of backing up

I promise you that if you ever lose any data stored either on your local computer or on your website, you'll instantly understand the importance of backing up. I'm probably saying this way too often, but the fact is that a hard disk failing is not a matter of *if*, it's a matter of *when*.

All it actually takes is a glitch in your server or a hacker's infection. And although most web hosts offer some kind of backing up services, it's good to have your own additional system set up (just in case). There are a couple of approaches to dealing with backups, which I will outline in the following sections.

## Easy, quick, and frequent content backups

The most important part of your site, and also the part that you will never be able to re-create is the content contained in the database. You should back up your database frequently. Exactly how often, will depend on the number of times you change your content.

If you're running a blog, do a backup whenever you've posted two or three new posts. If you're running a non-blog website, backup every time you make significant changes to your content or add new pages.

The question you should ask yourself is, "If my server or host completely fizzles out today, how much time would it take me to re-create what's not already backed up?"

Luckily, your website content is pretty easy to back up. You can directly export the content of your database using phpMyAdmin or any other database tools provided by your host.

You can also install the **Online Backup for WordPress** plugin that I described in *Chapter 5, Plugins and Widgets*. It's easy to use and will help you to back up not only the database but also the filesystem of your site (all files, uploads, themes, plugins, and other PHP code). The installation process of the plugin is 100 percent standard, so it doesn't require any additional explanation.

## Backing up everything

In addition to your database, there are other irreplaceable files that make up your WordPress website. These include:

- The theme you are using
- The plugins you've installed and activated
- The files you've uploaded
- The code you've changed by hand

WordPress stores all of these things (except for the last one, since you can modify any file you wish) in the same folder named `wp-content`. Every time you change your theme and install a new plugin, you should make sure that you have a backup of these things on your home computer. After that, you don't need to back these two things up regularly because they won't change.

However, the files you upload (for example media files, images, audio, icons, and so on) are a collection that changes over time as you add more files. If you add a photo with each blog post, then that collection changes as frequently as you post. You should be sure to create a backup of the `wp-content/uploads/` folder pretty regularly.

This you can do manually via FTP or through the Online Backup for WordPress plugin mentioned previously. When dealing with FTP, it's good to have an FTP program with a **synchronize** feature, and then you won't have to constantly re-download older files, or do a lot of hunting and pecking for new ones.

In the end, using Online Backup for WordPress is still a lot simpler and more user-friendly than playing with FTP.

## Verifying your backups


Be sure to verify your backups! Every time you download a database export, use FTP to download files, or get a backup package from Online Backup for WordPress, make sure to take a close look at the downloaded files. Sometimes backups get interrupted or there are glitches in the system. It's better to find that out right in the beginning, rather than when you need to rely on your backups later.

One more security feature worth pointing out inside Online Backup for WordPress is encryption. You can take any backup file and encrypt it using military-level algorithms. This will make the backup accessible only to the person who knows the encryption key (that is you). By doing so, you're making sure that no one will be able to hijack your backup and put any malicious code in it.

You can set this by going to **Tools | Online Backup | General Settings**:

Overview | Backup | Activity Log | Decrypt Backup | Schedule | **General Settings** | Online Backup Settings | Help & Support

### Encryption

 We highly recommend that you enable encryption. Using encryption will mean that nobody can access your backup files without first providing the encryption key.

DES is the lightest form of encryption, and actually uses more server resources than AES encryption. Only use DES if you need encryption but need it to be weak. AES encryption is the better encryption. The larger the number after it, the more server resources required to encrypt the data, but the better the protection provided. We recommend AES128 - it has the best balance in not being too resource intensive and still offering enterprise grade protection.

Encryption type:

Encryption key:  used to encrypt your backups - you should set it with numbers and letters or symbols.

Remember to write it down!

Show encryption key

*Your encryption key is just like a password, it can be anything you want it to be.*

## Getting a managed solution

There are various ways of getting your site safely backed up, and one of them is to sign up to a managed solution that handles the process for you with no supervision required. One of the popular services like this is called **CodeGuard** (get it at <https://www.codeguard.com/>). There's a free trial available.

CodeGuard handles your site backups automatically, based on a schedule. Every day, it analyzes your site, backs up every new entry in the database and every file in your site's file system, and then notifies you if it finds any changes. This allows you to keep your finger on the pulse and act fast if there's anything suspicious going on. Like, for example, when your files change but you are not the person making the changes. This might mean a hacker attack. CodeGuard also offers a time-machine-like functionality. This feature enables you to restore your site to one of its previous versions in case anything goes wrong and your site stops working.

## Upgrading WordPress

In *Chapter 2, Getting Started*, we briefly discussed upgrading your existing WordPress version to the latest available version. In this section, we will take a closer look at the upgrade process.

So what about the built-in upgrader? As of WordPress 2.6, you can upgrade in one click from within WordPress. You can use the built-in upgrade to replace steps 3 to 7 in the following list, but you should still carry out the other steps. Also, the built-in upgrade does not work on all servers.

## Upgrading from a very early version

Something to keep in mind is that if you're upgrading from a very early version of WordPress to a very recent version, you should do it in steps. That is, if you find yourself in a situation where you have to upgrade across a large span of version numbers, for example from 2.2 to 3.7.1, I highly recommend doing it in stages. Do the complete upgrade from 2.2 to 2.3, then from 2.3 to 2.5, then from 2.5 to 2.7, from 2.7 to 3.0, from 3.0 to 3.1, and so on until you reach 3.7.1. When doing this, you can usually do the full content and database backup just once, but verify in between versions that the gradual upgrades are going well before you move onto the next one.

You can download the previous stable versions of WordPress from the following page: <http://wordpress.org/download/release-archive/>.

Of course, another option would be to simply do a new installation of the latest version of WordPress and then move your previous content into it, and I encourage you to consider this course of action. However, sometimes content is harder to move than it is to do the upgrades; this will be up to you to decide your specific server situation and your comfort level with the choices.

## Steps for upgrading

The steps involved in upgrading WordPress are as follows:

1. Back up your database.
2. Back up your WordPress files.
3. Put WordPress in Maintenance Mode.
4. Deactivate all your plugins.
5. Download and extract WordPress.
6. Delete the old `wp-includes` and `wp-admin` folders. Be careful though. Don't delete your existing `wp-content` folder, or any files in it.
7. Upload the new files, which will overwrite the old files.
8. Run the WordPress upgrade program.
9. Update permalinks and `.htaccess`.
10. Install updated plugins and themes.

Each step has been described in detail in the following sections of the chapter.

### Backing up your database

Before upgrading WordPress, you should always back up the database. If anything goes horribly wrong with your upgrade, you won't lose everything. We reviewed how to back up your WordPress website earlier in this chapter, so you can refer back for specific instructions.

### Backing up your WordPress files

Remember, the complete content of what creates your site is contained not only in the database, but also in certain files on the server. Always back up all of your files as well, just in case something goes wrong with the upgrade.

Again, refer back to the backup instructions above if you need to review the steps.

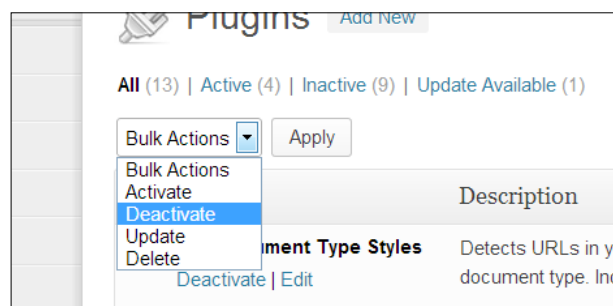
### Put WordPress in Maintenance Mode

If you think that visitors or other users will want to visit a WordPress site during the time you're upgrading it, you might want to consider putting the site into Maintenance Mode. This can be done relatively easily with a plugin like the following one: <http://wordpress.org/extend/plugins/themefuse-maintenance-mode/>. We already discussed it in *Chapter 5, Plugins and Widgets*, so please review it to get the full how-to. Basically, it will show a splash page to non-administrative users while you're working on the site, and you can even set it to be active for only a specific period of time.

## Deactivating all your plugins

The plugins that you have installed with your current version of WordPress may not work with the newest version. Also, if you leave them active while upgrading, your new WordPress installation may break. So, before upgrading WordPress, you should play it safe and deactivate all the active plugins. If you're using the built-in upgrader, you don't have to worry about this step because the upgrader does it for you.

To deactivate plugins, log into your WP Admin and navigate to **Plugins**. Deactivate everything at once by clicking the master checkbox at the top of the table, choosing **Deactivate** from the **Bulk Actions** menu, and clicking on **Apply**.



## Downloading and extracting WordPress

Now, download the WordPress ZIP file onto your computer from <http://wordpress.org/download/>.

After downloading WordPress, extract all of the files in a new folder named `wordpress`.

## Deleting old files

Now delete all the files and folders of your previous WordPress installation except these:

- `wp-config.php`
- `wp-content/`
- `wp-includes/languages/` (if you have used a specific language pack)
- `.htaccess` (if you have used custom permalinks)

The easiest way to delete the files and folders that you will be replacing is to access your server via FTP.

## Uploading the new files

If you're not already connected via FTP, connect now. Select all of the files in the `wordpress` folder on your computer, except for the `wp-content` folder, and upload them to your server.

Finally, give your `wp-config.php` file a closer look. Compare it against the new `wp-config-sample.php` file just to see if any new settings have been introduced. If so, you'll have to include them in your old `wp-config.php` file too.

## Running the WordPress upgrade program

WordPress takes care of the next step for you: running the upgrade. This script usually takes a look at your database and makes alterations to it, so that it is compatible with the new version of WordPress.

To access it, just point your browser to your WordPress website, and you'll be prompted to do the upgrade. Alternatively, you can take a shortcut by going directly to `http://yourwordpresssite.com/wp-admin/upgrade.php`.

Click on the **Upgrade WordPress** link.

## Updating permalinks and `.htaccess`

You may have to update the permalink settings so that they match the previous installation. Your permalink settings dictate what the `.htaccess` file should look like. If WordPress cannot access your `.htaccess` file because of permissions problems, the permalinks page will display a message letting you know about it. That message will also tell you what text needs to be in the `.htaccess` file, so that you can create or update it yourself.

Lastly, if you've previously used a plugin like BulletProof Security (available at `http://wordpress.org/extend/plugins/bulletproof-security/`) to handle your `.htaccess` file and then enabling the plugin back on after the upgrade should give the `.htaccess` file a correct structure again.

## Installing updated plugins and themes

In your WP Admin, visit the plugins page again. If there are new versions of any of your installed but now inactive plugins, there'll be a note telling you so. If you have any plugins that are not part of the WordPress **Plugin Directory**, this is a good time to check the websites for those plugins to see if there's an upgrade available.



You can also take a look at the **Plugin Compatibility** lists on this page:  
[http://codex.wordpress.org/Plugins/Plugin\\_Compatibility](http://codex.wordpress.org/Plugins/Plugin_Compatibility).

Once you're sure that the plugins you want to use are up-to-date, activate them one at a time so that there are no problems.

This is also a good time to check for updates for the theme you are using. In the `wp-content` folder on your computer (which you did not upload, remember?) there is an updated version of the default theme `twentythirteen`, which I always like to have around even if I'm not using it. You can simply replace the entire `wp-content/themes/twentythirteen` folder with the new one.

You can check for a new version of the theme you are using on the developer's website, or in the WordPress **Theme Directory**. Of course, you have to be sure you haven't made any theme customizations directly to the theme you're using. If you have, they will be overwritten when you run the theme upgrade. If you want to customize an existing theme, be sure to make a child theme (covered in an earlier chapter of this book).

## Migrating or restoring a WordPress site

Sometimes, you may find yourself in a situation where you need to move your WordPress website from one server to another or from one URL to another. Alternatively, if something gets fried on your server and is restored, you need to re-create your damaged WordPress site. Here, you'll essentially need to do the same things as you would in a migration.

I highly recommend that you check out this page in the WordPress codex, which has detailed step-by-step instructions to migrate your WordPress website under a variety of different circumstances: [http://codex.wordpress.org/Moving\\_WordPress](http://codex.wordpress.org/Moving_WordPress).

This page will be kept up-to-date as time moves on. If you need to do a migration and don't have access to the codex right now, you can follow these steps for migration:

1. Download a backup of your database (as described earlier in this chapter, in the section on backing up). If your URL is going to change, you may want to use a different plugin to download your database. It's called WP Migrate DB: <http://wordpress.org/extend/plugins/wp-migrate-db/>. This plugin will change URLs for you.
2. Download all of your files (as described earlier in this chapter, in the section on backing up).

3. Look in your downloaded files for `wp-config.php`. Find the lines that define the connection to the database.  
Edit these lines so that they now have the database name, database username, database password, and database hostname for your new database.
4. Upload all of your files to your new server.
5. Implement your SQL file in your new database.
6. Change the permissions of your `wp-content` folder if necessary, so that you'll be able to upload files without any problems.
7. Change the absolute path of your WordPress folder in the database. You can do this by getting a custom script like <http://interconnectit.com/products/search-and-replace-for-wordpress-databases/> – please follow the URL for a step-by-step tutorial.
8. Log in to your new WP Admin, and check the permalinks. You may have to reset them if your `.htaccess` file didn't come over properly.

You're done!

If you're restoring your site on the same server, with no changes in the location or the database, then you can skip steps 3 and 7. Steps 1 and 2 (to back up) should be done before the meltdown!

## **Acting in case of a site crash**

This is quite unfortunate, but it does happen from time to time. One day, you might wake up and find your site not working, just like that. Getting it back on can take a while depending on the cause of the crash. Here are some of the most common reasons for a WordPress site crashing:

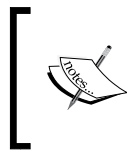
- Bad hosting service running the site
- Poor quality themes or plugins with badly structured code
- Weak passwords for admin accounts (prone to hacker attacks)

Here's my routine when dealing with a site that has just crashed.

What's important is to check if your site has started working again after handling each step. Most of the time you don't have to go through all of them, your site will probably start working again somewhere in the middle of the process:

1. Deactivate or even delete all plugins. Actually, you don't have to delete anything, just rename the main `plugins` directory, so WordPress won't be able to find it.
2. Rename the current theme folder, so WordPress goes back to the default theme.
3. Take a working backup copy of the site and restore all contents from it.
4. Overwrite all of your WordPress core files with new ones. Proceed just like you'd want to upgrade WordPress manually (described earlier in this chapter).

If your site has started working again then you can begin reactivating your theme and your plugins one by one. Check if the site is still working after each activation.



To get a more detailed guide on how to restore WordPress after a crash feel free to visit the following post: <http://themefuse.com/blog/restoring-wordpress-after-a-crash/>.

## Setting file permissions

To install and maintain WordPress properly, you may need to change permissions to different files and folders in the WordPress folder so that uploads and built-in updates will work from within WordPress. Usually, this affects people on Unix servers, though it affects some Windows servers as well.

## Explaining file permissions

File permissions are settings that indicate who is allowed to do what. That is, some users may have permission to alter the contents of a file, some may have permission to only read it, and some may not even have read/write access. In addition to read/write permissions, there are also "execute" permissions. If a file is executable, then this permission indicates who can execute that file.

For Unix file permissions at a glance, look at the following table:

File/Folder	Owner permission	Group permission	User permission	Total	Numerical equivalent
/	rwX	rw	rw	rwXrw-rw-	766
/.htaccess	rwX	rw	rw	rwXrw-rw-	766
/wp-admin	rwX	r--	r--	rwXr--r--	744
/wp-includes	rwX	r--	r--	rwXr--r--	744
/wp-content	rwX	rwX	rw-	rwXrwXrw-	776
/wp-content/ themes	rwX	rwX	rw-	rwXrwXrw-	776
/wp-content/ plugins	rwX	rw-	rw-	rwXrw-rw-	766

## Permissions for WordPress

The best permission scheme for a WordPress installation is for all of the files to be owned by your user (your server's user, not your WordPress user), and be writable by your user. On top of that, any file that WordPress will need to modify (such as just about anything in `wp-content`) should be group-owned by both your user and the webserver's user (often called **dhapache** or nobody).

If you've installed your own WordPress website yourself, you shouldn't need to modify any permissions for all WordPress functionality to work.

If you're having trouble using the built-in upgrader or plugin installer, do not `chmod` everything to `777` (world-writable). Instead, check with the hosting provider or people who run your server, and ask what they recommend.

## How to set permissions

You can change permissions to files and folders using any FTP client. If you have shell access, you can use shell commands to change file permissions. If you're using an FTP client, select the files you want to change permissions for and look for menus such as **Get Info**, **File Attributes**, or **Change Permissions**. There will be a GUI, often with checkboxes, that lets you choose permissions for different files. Some hosting control panels, such as Fantastico for example, allow you to change permissions through the control panel itself. If you are using shell, change the file permissions with the `chmod` command.

For example, the `wp-admin` folder should be set as `rwxr--r--` or `744`; and to change the permissions for the `wp-admin/` folder, run the following command:

```
chmod -R wp-admin 744
```



Learn more about WordPress and file permissions at [http://codex.wordpress.org/Changing\\_File\\_Permissions](http://codex.wordpress.org/Changing_File_Permissions).

## Troubleshooting

In this section, we will discuss problems that may arise during installation and execution of WordPress, and provide solutions for troubleshooting them.

### Troubleshooting during installation

Most of the problems discussed here have been taken from the WordPress installation FAQs (**Frequently Asked Questions**) and **Troubleshooting FAQs**.

#### Headers already sent

The description of this problem is as follows:

- **Problem:** When you point your browser at your website, you may get an error that displays a **headers already sent** message on your page. The whole page may look scrambled, and it will not function.
- **Cause:** WordPress uses PHP session functions. If anything is sent by the server to the browser before these session functions, even if just a blank space, then the session functions will not work properly.
- **Solution:** You have to figure out where the error lies. Usually, it is a file that you have edited manually. If you remember, you edited the `wp-config.php` file while installing WordPress. Open the file with your text editor, and make sure that there is nothing before `<?php` in the first line or after `?>` in the last line.

## Page comes with only PHP code

The description of this problem is as follows:

- **Problem:** When you point your browser at your website it displays the PHP code instead of its contents.
- **Cause:** This happens when your server is not parsing PHP, but is instead treating it the same as any text or HTML file. This is a server configuration problem; either PHP is not installed on your server or it is not configured to function properly.
- **Solution:** To solve this problem, contact the system administrator for your server or try installing PHP.

## Cannot connect to the MySQL database

The description of this problem is as follows:

- **Problem:** WordPress cannot connect to the MySQL database, and is displaying an error.
- **Cause:** This might happen if:
  - The database parameters are incorrect
  - The daemon/service is not running properly
  - Your server is running an outdated version of MySQL
- **Solution:** To solve this problem, you can try the following:
  1. Open your `wp-config.php` file, and check that the database parameters are correct.
  2. If you are sure that these settings are correct, check if the MySQL daemon/service is running properly. If MySQL is not running, run this service. If MySQL is already running, try restarting the service. If you are not running your own server, check in with your hosting company's support people.
  3. If you are sure that your database parameters are fine, and MySQL is also running, connect to MySQL using your MySQL command-line tool and run the following commands:

```
set password = OLD_PASSWORD('your_current_password');  
flush privileges;
```

This will use the old encryption of passwords so that PHP can connect to MySQL.

---

## Basic troubleshooting

As you have probably already figured out, the best place to look for troubleshooting tips is the `WordPress.org` website – both the codex and the support forum. The codex even has a page devoted to basic troubleshooting: <http://codex.wordpress.org/Troubleshooting>. There's also an updated Troubleshooting Master List for each new version of WordPress. The newest one is located at <http://wordpress.org/support/topic/wordpress-37-master-list>.

Below are some of the most common problems people encounter when setting up WordPress; if you don't see yours here I encourage you to visit the codex.

### Cannot see posts

The description of this problem is as follows:

- **Problem:** Posts are not seen, and the message "search doesn't meet criteria" is displayed.
- **Cause:** This can happen because of caching. For example, you have searched once, and WordPress stored the search result inside its cache; so every time you visit the page, you see the old result.
- **Solution:** You can solve this problem by clearing the cache and cookies from your browser.

### I don't receive the e-mailed passwords

The description of this problem is as follows:

- **Problem:** You don't receive the e-mailed passwords.
- **Cause:** This problem may happen if your web server has no **SMTP (Simple Mail Transfer Protocol)** server installed, or if the mail function is explicitly disabled.
- **Solution:** Please contact your system administrator, or try installing Sendmail (or any other mail server) properly.

## Tips for theme development

In *Chapter 7, Developing Your Own Theme*, we covered theme development pretty thoroughly, though you can get a much more in-depth tutorial in theme development from the excellent book *WordPress Theme Design*, Packt Publishing.

This section lists the top template tags and stylesheet classes that you'll want to have if you're going to be developing themes. These are the most essential (with some of my personal favorites thrown in).

## Template tags

Following is a list of the most used template tags. For a complete list, visit the Codex. This is a good place to start: [http://codex.wordpress.org/Template\\_Tags](http://codex.wordpress.org/Template_Tags).

You can also have a look into the Function Reference sheet at [http://codex.wordpress.org/Function\\_Reference/](http://codex.wordpress.org/Function_Reference/).

In the list that follows, I do not cover the parameters that can be passed to these functions. You'll want to visit the Codex to find out about the default settings for each tag and how to override them. The header and informational tags are as follows:

---

The tag	What it does
<code>wp_title()</code>	Prints an appropriate title for your blog (the post title, the archives title, the page title, or whatever is appropriate for the current page)
<code>bloginfo('name')</code>	Prints out the name of your blog, as specified on the main options page in your WP Admin. You can also use a similar tag— <code>get_bloginfo()</code> , which returns the information instead of displaying it
<code>wp_head()</code>	An essential part of the <code>&lt;head&gt;&lt;/head&gt;</code> tag, because a variety of things get printed out by this tag, depending on the details of the blog
<code>bloginfo('stylesheet_url')</code>	Prints out the path to the stylesheet of the current theme
<code>bloginfo('rss2_url')</code>	Prints out the RSS 2.0 feed URL for your blog
<code>body_class()</code>	Prints out a list of appropriate class names in the body tag. It should be used to replace <code>&lt;body&gt;</code> like the following:  <code>&lt;body &lt;?php body_class(); ?&gt;&gt;</code>

---

The following tags can be used inside the loop:

---

The tag	What it does
<code>the_title()</code>	Prints out the title of the current post or page
<code>the_time()</code>	Prints out the date and time of the post or page
<code>the_content()</code>	Prints out the formatted post or page content
<code>the_category()</code>	Prints out a list of the categories that belong to this post
<code>the_tags()</code>	Prints out a list of the tags associated with this post
<code>the_author()</code>	Prints out the name of the post or page author

---



The tag	What it does
<code>edit_post_link()</code>	If the person viewing the blog is a logged-in blog user, this tag will print out a link for editing the post (very handy!)
<code>the_permalink()</code>	Prints out the URL of the post or page itself (must be used within a <code>&lt;a href=""&gt;&lt;/a&gt;</code> tag)
<code>comments_popup_link()</code>	If <code>comments_popup_script</code> is not used, this displays a normal link to the comments for the post or page
<code>post_class()</code>	If you put this tag inside the <code>&lt;div&gt;</code> for your posts, it will generate a list of classes for the categories and tags that belong to this post. For example, if you put the following in your template: <pre>&lt;div &lt;?php post_class(); ?&gt;&gt;</pre> WordPress will print something like the following: <pre>&lt;div class="post category-recipes category-locavore tag-holiday tag-pasta tag-recipe tag-spinach"&gt;</pre>
<code>get_post_meta()</code>	Use this function to get the value stored in a custom field. Just pass this function the current <code>post id</code> , the name of the custom field you want, and <code>true</code> to get the value of that custom field
<code>get_the_post_thumbnail()</code>	Prints out a complete <code>img</code> tag for the featured image

The following tags can be used for lists and navigation:

The tag	What it does
<code>previous_post_link()</code>	When viewing a single post, this prints a link to the previous post (the one with the next newest timestamp)
<code>next_post_link()</code>	When viewing a single post, this prints a link to the next post (the one with the next newer timestamp)
<code>wp_list_pages()</code>	Prints a list of all the pages in your WordPress site
<code>wp_get_archives()</code>	Prints a list of archives (by post, by month, and so on)

The following tags can be used to include PHP files:

The tag	What it does
<code>get_header()</code>	Includes <code>header.php</code> from the current theme folder
<code>get_footer()</code>	Includes <code>footer.php</code> from the current theme folder
<code>get_sidebar()</code>	Includes <code>sidebar.php</code> from the current theme folder
<code>comments_template()</code>	Prints the standard list of comments and comment-submission forms, unless there is a file in the theme folder named <code>comments.php</code> , in which case that is included instead
<code>get_search_form()</code>	Prints the standard search form
<code>include(TEMPLATEPATH. '/filename.php')</code>	Includes <code>filename.php</code> from the current theme folder

The following are some of the most useful conditional tags. Note that some of them *can* take a parameter, so be sure to look them up in the codex for details.

The tag	What it does
<code>is_front_page()</code>	Returns <code>true</code> if user is viewing the front page of the site, regardless of whether it's the most recent blog post or a page
<code>is_home()</code>	Returns <code>true</code> if user is viewing the main page of your blog, which can either be the front page of your site or the page you designated as the Posts page in <b>Settings   Reading</b>
<code>is_page()</code>	Returns <code>true</code> if user is viewing a page
<code>is_single</code>	Returns <code>true</code> if user is viewing a single post
<code>is_archive()</code>	Returns <code>true</code> if user is viewing an archive page of blog posts (monthly, yearly, category, tag, and so on)
<code>is_search()</code>	Returns <code>true</code> if user is viewing search results
<code>has_post_thumbnail()</code>	Returns <code>true</code> if the post (only parameter is <code>post ID</code> ) has a featured image or designated thumbnail assigned to it

Learn more about conditional tags at [http://codex.wordpress.org/Conditional\\_Tags](http://codex.wordpress.org/Conditional_Tags).

## Class styles generated by WordPress

WordPress helpfully applies classes to just about everything it generates, thus making it easy for you to style WordPress-generated elements on your page.

Here is a starter list of those styles. If you want to know what the other styles are, create a template and view the source of the page it creates.

Class or ID	Where to find it
<code>.page_item</code>	On the <code>&lt;li&gt;</code> tag of every page in the generated page list
<code>.current_page_item</code>	On the <code>&lt;li&gt;</code> tag of the current page in the generated page list
<code>.current_page_parent</code>	On the <code>&lt;li&gt;</code> tag of the parent of the current page in the generated page list
<code>.page-item-23</code>	On the <code>&lt;li&gt;</code> tag of the page with ID = 23 (there is one of these for each page) in the generated page list
<code>.menu-item</code>	On the <code>&lt;li&gt;</code> tag of every item in the generated nav list. As with pages, <code>menu-item-parent</code> , <code>current-menu-item</code> , and so on are also generated for appropriate items.
<code>.widget</code>	On the <code>&lt;li&gt;</code> tag of every widget
<code>.cat-item</code>	On the <code>&lt;li&gt;</code> tag of every category in the generated category list
<code>.current-cat</code>	On the <code>&lt;li&gt;</code> tag of the current category in the generated category list
<code>.cat-item-13</code>	On the <code>&lt;li&gt;</code> tag of the category with ID = 13 (there is one of these for each category) in the generated category list
<code>#searchform</code>	On the <code>&lt;form&gt;</code> tag for the generated search form

## Learning more

If you want to learn more about how WordPress deals with CSS and various style classes, please visit either of the following resources:

- The official guide on CSS in WordPress at <http://codex.wordpress.org/CSS>
- Default WordPress Generated CSS Cheat Sheet for Beginners at <http://www.wpbeginner.com/wp-themes/default-wordpress-generated-css-cheat-sheet-for-beginners/>
- WordPress Default CSS Styles at <http://digwp.com/2010/05/default-wordpress-css-styles-hooks/>

## Summary

In this chapter, we covered many of the common administrative tasks you may face when you're managing a WordPress-driven website. This includes backing up your database and files, moving your WordPress installation from one server or folder to another, and doing general problem-solving and troubleshooting. We also covered some of the most basic and useful template tags that you'll need when creating your own WordPress themes.

You should now feel well-equipped to address all of the more and less usual administrative tasks for your website or blog.

WordPress is a top-notch CMS, which has matured tremendously over the years. The WordPress Admin panel is designed to be user-friendly, and is continually being improved. The code that underlies WordPress is robust, and is the creation of a large community of dedicated developers. Additionally, WordPress' functionality can be extended through the use of plugins.

I hope you have enjoyed this book, and have gotten a strong start with administering and using WordPress for your own site, whatever it may be. Be sure to stay connected to the WordPress open source community!