Chapter 1

WordPress As a Professional Web Development Tool

WordPress was originally established as a tool for bloggers, aimed at the growing community of people writing personal, technical, or business blogs, and providing them with a platform they could use to host that blog on their own server. But WordPress has evolved—significantly. It is no longer simply a blogging tool, but a fully fledged content management system (CMS), with a myriad of features that enable developers to experiment with the structure and functionality of a site, customize the dashboard and admin screens for users, and install plugins to enable whatever additional capabilities the site needs.

WordPress, to put it simply, is now a professional web development tool, used by thousands of web professionals to build sites for themselves, their clients, and other users. It's a tool on which you can build a business.

This chapter looks at the WordPress features you can harness as a professional web developer, and identifies how your working practices may need to change if you're scaling up your WordPress practice. You'll learn some techniques for improving your working and coding practices when collaborating as part of a larger team, and find out how to manage large web design and development projects, including the skills you'll need and the people you can expect to work with. You'll also look at the implications of building and possibly selling WordPress themes and plugins for release to other users and developers.

What It Means to Be a Professional WordPress Developer

If you're reading this book, there's a good chance that you already use WordPress on a professional or semi-professional basis. Maybe you work for an agency that builds client sites in WordPress, or for a company with a WordPress site that you maintain. You could be a freelance WordPress developer, or perhaps you're starting out as a fully fledged WordPress professional, setting up your own agency and building WordPress-powered sites for your own clients.

If you're going to do this professionally, you'll have to adapt your working style and practices, as well as your approach to development and coding. As a bare minimum, you'll need to do the following:

- Ensure that you understand WordPress well enough to build a diverse range of complex sites with it.
- Change the way you code so that people you're working with can understand what you've done and work with your code.

- Start thinking imaginatively about WordPress development, and in particular about how you can harness WordPress to solve your clients' real-world problems.
- Develop the skills needed to explain to your clients how WordPress, and the site you design and develop using it, can benefit them.
- Come to grips with the more commercial aspects of WordPress—using it to enhance your clients' SEO, seeing the potential to maximize your earning potential from WordPress, and possibly start selling themes or plugins.

As you work through this book you'll see that there is no one-size-fits-all approach to being a WordPress professional, but there are some practices and capabilities all WordPress professionals need, and others that will be more relevant to you depending on exactly how you work with WordPress. We'll be revisiting this theme throughout the book, particularly in Part IV, "Pushing the Limits: The Best Tools for Site Development."

Professional Coding Practices

The first thing you need to do if you want to scale up your approach to WordPress is review the way you code. Ask yourself: Who else looks at or works with your code at the moment?

The answer will vary according to where and how you work. If you're not running your own agency, you are probably not the person with ultimate responsibility for the quality and robustness of your code. Conversely, you might be the only person who works with your code.

Professional coding practices are about more than writing valid, standards-compliant code, although that is essential—and hopefully you already do this. It's about writing code that other developers can happily work with and develop further. If you're developing themes or plugins for other WordPress users to install, then you may need to focus on writing code that is resistant to the kind of hacking a WordPress novice might subject it to.

There are a few aspects to professional coding practices:

- Make sure your code is valid and standards-compliant.
- Use up-to-date coding methods.
- Comply with the WordPress coding standards.
- Make your code tidy.
- Structure your files well.
- Be consistent.
- Use comments liberally.

The following sections describe what these guidelines mean in practice.

Valid and Standards-Compliant Code

Yes, I've already said this, but it is absolutely fundamental. If you haven't run the HTML in your themes or plugins through a validator, do it! The most popular method for validating your code is to use the W3C validator at

`http://validator.w3.org`. This is the most widely used approach and the first place to start. However, validating your code involves more than just this. It includes (but is by no means limited to) the following:

- Validating against accessibility standards, including WAI, or the Web Accessibility Intitiative
- Checking links
- Validating feeds
- Cross-browser compatibility checking (including handheld and tablet devices)

For a long list of validation tools and techniques, take a look at the guidance on the WordPress codex at `http://codex.wordpress.org/Validating_a_Website`.

Up-to-Date Coding Methods

If others are going to be working with your code, especially if they're going to be paying for it, it's imperative that you write code that is up to date. For example:

- Don't use tables for layout (we really hope you stopped doing this a while back, but it bears repeating).
- Use the most recent versions of the main coding languages—HTML5 and CSS3.
- Avoid using deprecated code—although browsers are generally forgiving, your users may not be.
- Accept that you can't keep up to date with everything, but make sure you read web development blogs, journals, and magazines so you're not completely out of the loop.
- If a project involves something you haven't done for a while (or at all), do some research before starting—or hire a specialist as a freelancer or staff member.

WordPress Coding Standards

The WordPress codex details a set of standard coding practices, designed to help enhance consistency in WordPress code structure. This includes standards for PHP, HTML, and CSS.

Get to know these standards and use them. Even if you come across code that doesn't adhere to them, it's good practice to use them yourself and to expect members of your team to do so. The consistency and clarity that this brings to your code will help others who work with it, including your team—and your clients if you are selling themes or plugins.

Tidy Code

If other people are going to be working with your code, especially if they aren't advanced developers themselves, your code has to be easy to understand. Adhering to the following best practices will result in code that is easier to work with and harder to break:

- Use line breaks and indentation to help others see how your code is structured in one glance.
- Avoid empty divs and other elements added purely for styling—try to keep your markup lean and use CSS to style it, including the use of CSS pseudo-elements where appropriate.
- Rationalize your stylesheets to avoid duplication—if two or more elements or classes have the same styling, code it once instead of doing it repeatedly for each one.

Well-Structured Files

Files that have a clear structure are much easier for other developers to work with.

Your markup should be written in the order it appears on the page, even where you're using CSS to position it outside that flow. So if your layout shows the header first, then the content and sidebar followed by a footer, code it in that order. This will not only help other developers, but also improve accessibility, as screen readers read the code in the order in which it's written.

Structure your stylesheets in sections, with a summary of the structure at the beginning. For example, the Twenty Eleven theme (`http://wordpress.org/extend/themes/twentyeleven`) is split into no less than 21 sections, which include the following:

- CSS reset (this should always come first)
- Structure (for overall page layout)
- Global (for global elements such as fonts, list styling, and colors)
- Header (specific styling for the header layout)
- Menu (for the navigation menu)
- Content (for page or post content)
- Link (for links, including links in text and page titles)
- Image (for images within the content)
- Widgets
- Footer
- Media queries

Your themes may not need so many sections, but if you are developing themes for others to download and use, you'll need to cover all the bases.

Consistency

People who need to work with your code don't want any surprises, so consistency is essential.

For example, when coding HTML within PHP files, use a consistent method.

Some developers prefer to write opening and closing PHP tags every time they need to add markup, as shown in the following very simplified extract from the WordPress loop:

```
<?php if ( have_posts() ) while ( have_posts() ) : the_post(); ?>
  <article>
    <?php the_content(); ?>
  </article>
<?php endwhile; ?>
```

Other developers prefer to stay in PHP, and use `echo` instead:

```
<?php if ( have_posts() ) while ( have_posts() ) : the_post(); ?>
  echo "<article>";
    the_content();
  echo "</article>";
endwhile; ?>
```

The first approach is probably the most commonly used, and it is preferable if the people who will be working with your code are more familiar with HTML than they are with PHP. Whichever you choose, make sure you stick with it.

You should also use and format comments consistently. For example, you might prefer to add comments to your PHP on the same line as the code:

```
<?php the_content(); ?> // this is a comment
```

Conversely, you might prefer adding comments on separate lines:

```
<?php
/* this is a multiline comment
which spans more than one line */
?>
```

Again, whichever you do, be consistent and use the correct syntax.

CSS styles comments in the same way. HTML, however, uses only one kind of commenting syntax:

```
<article><!-- comments here --></article>
```

You already know how you prefer to code, and don't need us to tell you how to do it—but it's important to consider your preferences when other people will be working with that code, to ensure consistency and ease of understanding. You may also want to specify some coding practices and habits for other members of your team so you're all working in the same way.

Avoiding Duplication of Function and File Names

Whenever you write a new function, it's essential that you give that function a unique name to void any conflicts with WordPress functions or functions in other plugins or the active theme. It's normal practice to add a prefix to each function that corresponds to the name of your theme or plugin.

A function such as the following runs the risk of not being unique:

```
<?php
function name_of_function(){
 // function contents
}
?>
```

Instead, it should have a prefix, as follows:

```
<?php
function prefix_name_of_function(){
 // function contents
}
?>
```

In this book I prefix all functions with `wpptl_`:

```
<?php
function wpptl_name_of_function(){
 // function contents
}
?>
```

This also applies to any the names of any hooks you register and of files you create in plugins. It doesn't apply to theme template files—these need to adopt the standard filenames for template files if they're going to work.

Liberal Use of Comments

Still on the topic of comments, it's worth stressing how important they are when working with other developers. Try to anticipate any places in your code where a comment could be useful to someone unfamiliar with it. This will help other developers understand at a glance what your code is designed to achieve.

As I discuss later in this book, there is almost always more than one way to achieve something in WordPress, and in code. The next person to work on your files may have a different way they prefer to code something, so what you've done may not make sense to them. By adding a comment, you're telling them what your code does and where it applies.

Commenting liberally will also help you when you revisit some code you wrote a year ago on a client site that now needs updating. The chances are good that this will happen at 4:00 p.m. on a Friday when the client's site has hit a problem and you need to edit some offending code fast. Good commenting can mean the difference between finding the problem straightaway, making the edit, and being home in time for the weekend, versus slogging through thousands of lines of code and missing your kid's soccer game.

Following are some tips for good comments:

- Use comments at the beginning of your file to explain what the file does. For example, in template files include a note about what data it displays and any customizations you've made to the loop or other parts of the file; and in plugin files add a note regarding its functionality. For example:

```
/**
* Template Name:    sidebar-products.php
*
* The include file used for the sidebar on pages using the
  single-product.php template. Displays a gallery of product images
  (omitting the featured image which is displayed in the content).
*
* @package WordPress Pushing the Limits
* @since 1.0
*/
```

This comment tells users the name of the template file, what it does, the theme it is part of (`@package`), and which version of the theme it has been in place since (`@since`). You should use a similar system for plugin files.

- Use comments to demarcate the sections of your code, not only in stylesheets but also in your theme template files and functions file.
- Comment anything that is nonstandard.
- Comment anything that took you a while to work out—use detailed comments so when you come back to it, you don't have to think it through again.
- Comment anything that you know someone else will be working on—for example, if your theme files contain scripts that you'll be asking a JavaScript developer to perfect, add comments explaining where they apply in the site.
- Use wording in your comments that you can find using a search later—so don't abbreviate, and use terms others would understand.
- Whenever you comment out some code, add a comment to yourself containing the reason. That way, if you forget to remove the code after you've finished, or want to add it back in the future, you'll know what's going on.
- When in doubt, add a comment!

Approaching Large-Scale Site Design

So, now that you're running a professional WordPress business (or planning to), you're going to want to attract big clients—clients with exciting, complex requirements for their website, and the budget to match.

The reality, of course, is that not all of your clients will be like this—indeed, if you're new to this game, the chances are good that smaller clients will be your bread and butter in the early years, but it's a good idea to set your sights high and develop working practices in line with this goal.

You may have worked on large, complex sites before, perhaps as an employee in an organization's web team. However, you may not have had overall responsibility for the site, for managing its development, and for working with its stakeholders.

To work with clients and manage the completion of large-scale web design projects, you need some key skills:

- The ability to work with a range of stakeholders
- A balance between technical capability and business sense
- Communication skills
- Planning and project management skills
- Problem-solving skills

The following sections take a look at how each of these skills applies to large-scale site development.

Working with a Range of Stakeholders

When managing large-site builds, you typically work with a range of stakeholders who have hugely varying expectations and different levels of understanding regarding what you're trying to achieve. Some stakeholders you might need to work with include the following:

- **The client's project manager or website manager.** This may well be the person who commissioned the work (but may not have approved it). He or she will have a sufficient level of technical capability but may know little or nothing about WordPress, especially if the client is moving to WordPress for the first time. This individual is an important ally for you as someone with an interest in the project's successful completion but may not have the authority to make important decisions.
- **The client's senior managers or managing director.** These people will be less closely involved in the project but have the power to make key decisions, including pulling the plug on your project or drastically changing its direction. Not involving these individuals at key stages in the project can be a big risk—if you complete most of the work before you present the website to them and they don't like it, you may have to start again from scratch.
- **The client's technical staff or website editors.** The client may have employees whose job it will be to maintain and edit the site once it is launched—unless they require you to do this. If not, you may have to train these employees to use WordPress and/or work collaboratively with them on some of the code. They are unlikely to be decision-makers.
- **Your own team.** You may have one or more partners in your agency, and a team of developers and designers, either freelance or employed by you. Your role is to ensure that they know their role in any project you're managing, they have the information and resources they need to do the work, they get feedback on that work, and they know what the deadlines are. You also need to oversee the quality of their work and intervene if it is not up to scratch.

As you can see, you could be working with some very different people; therefore, in order to ensure that your projects are successful, you'll likely have to adopt a different approach with each group—something you may never have done before if you're used to spending most of your time on development. You'll need to use the set of skills described in the following sections to make it work.

Balancing Technical Capability and Business Sense

Technical capability is essential. If you weren't already proficient in WordPress, you wouldn't be setting up a WordPress business. But you also need business sense, which comes into play not only when you're dealing with clients, but also when you're managing your own business. Obviously, it's beyond the scope of this book to teach you how to manage your accounts, form a business plan, and so on—you can find hundreds, if not thousands, of resources to help you with that—but it's worth noting that having or developing a business sense will help you choose the right WordPress projects and make a success of them.

When you're talking to a client about their website build, it's important to be able to step back from the details of a project to see the bigger picture—so you can translate technical knowledge into something that has a tangible benefit. For example, you may be great at building responsive sites, but if you can't explain the benefits to your client, they won't pay you to do it.

Communication Skills

"Code is poetry," or so the saying goes, and it is—well-written code can make a website soar, in terms of content, functionality, and performance. However, most clients don't understand code—and they probably don't

understand web development or WordPress, at least not as well as you do. They may have some knowledge of web design and some fixed ideas about what their site should look like, and some of these may be good—we would caution against challenging the client on every design decision on the basis of your superior experience. Some of them may not be so good, however, and you'll need the communication skills to explain why—and then work together to find a resolution that will meet the client's needs.

Following are some of the scenarios in which you need to ensure effective communication with clients and your team:

- **Client pitches.** If you're lucky, all your work will come from referrals and you'll never have to do this, but most agencies need to do a client pitch at some point. Even if you are the only agency in the running, you still have to convince the client to hire you. At these meetings you need to be able to explain what you can offer in terms that the client understands and can identify with—in other words, a combination of business benefits and enough technical background to convince them of your expertise.
- **Project and planning meetings.** After landing a project, you will be working closely with the client to identify their requirements and determine how to implement them using WordPress. It's important to fully understand the requirements before diving straight into solutions, to avoid going down the wrong track. You need to both be a good listener and ask the right questions in order to extract the relevant information and probe for more detail. You should also be sure you confirm your understanding with the client—preferably in writing after meeting, to avoid any potential disagreements later.
- **Dealing with disagreement.** If you work well with your client and have a collaborative style, disagreements should be rare, but they do happen to most web designers at some point or other. If your client disagrees with something you've done, try not to get defensive. Remember that the site is theirs, and that they understand their business and its needs better than you do. If you don't think their idea will get the best results for the project or their business, explain it in those terms—"This is what I think will achieve the most conversions for your website and this is why," rather than "I've designed hundreds of websites and I know what works." For a definitive guide to working well with clients and avoiding conflict, you could do a lot worse than reading Paul Boag's *Client Centric Web Design*, or see his blog at `http://boagworld.com`.
- **Managing your team.** If you are working with a team of designers or developers, whether you personally employ them or not, you need to effectively communicate with them. At a minimum they need to know the following:
  - What you are hiring them to do
  - Your expectations in terms of quality and working style (e.g., how they code, what browsers or devices they should use for testing)
  - Deliverables and deadlines
  - The consequences of not delivering

  It's important to build a solid relationship with your team if you want them to give you their best work—especially when the team is made up of your own employees, who you want to feel a real sense of loyalty toward your agency. Part of your management effort may also include occasions when they don't deliver work on time or to standard, or when they're unavailable.

There is a reason why you can find plenty of resources out there to help you manage and communicate with a team: It's crucial to the success of your agency. We've listed some of these, along with other relevant resources, in the "Additional Resources" section at the end of this chapter.

Planning and Project Management Skills

This is a biggie! If you will be managing large-scale web development projects, and hopefully more than one at a time, you need to plan thoroughly and hone your project management skills.

The strongest technical skills and the smoothest relationships with your clients and your team won't offset a lack of planning and effective project management. Managing large projects with teams of developers means spending a lot of your time doing the following:

- **Planning projects.** Large-scale web development projects need to have a project plan. Depending on your preferences and experience, you have a few options for creating this: project management software, a simple spreadsheet, or an online project management and collaboration tool. Whatever you choose, you need a plan that at the very least specifies the following:
  - What needs to be done
  - Who needs to do it
  - When it must be completed

  In addition, you might include dependencies—stages of a project that can't start until another one is compete (for example, you can't add content to a site until your client has provided it).

  Make sure all stakeholders, including everyone with tasks to perform (and their managers if relevant), have provided their input to the plan and are committed to completing their actions within the deadlines agreed upon. Don't just sit down and write a plan yourself and expect everyone to stick to it—check everyone's availability before selecting any deadlines.
- **Monitoring progress.** As the project progresses, you need to keep on top of things. Check progress against deliverables and milestones on a regular basis—not just your own actions. If a delay seems likely, talk to the people involved to determine what can be done to get back on track. Do this as early as possible to avoid renegotiating deadlines when milestones aren't met because problems were communicated with only a few days left to deal with them.
- **Communicating progress to stakeholders.** Make sure anyone involved with the project knows what is expected of them next and what they can expect from you. This applies particularly to the client, who is likely to be focused on other things—for example, when sending the client work for review, specify the date by which you expect their feedback.

  If you suspect that the client may struggle to complete necessary tasks, make success more likely by scheduling time to work with them on site. For example, schedule meetings to plan the site's content together and agree how it will be structured, rather than expect them to do it in a vacuum. Most clients will appreciate your expertise about planning website content and find that this starting point makes it easier for them to complete the work.
- **Dealing with problems or delays.** Most large website builds come up against a problem at some point. Try to spot problems early and have the courage to deal with them, rather than hope they'll go away. That way, they'll remain small problems instead of turning into messy, large ones.
  - If you think you or your team won't be able to meet a deadline, deal with it early—either find a way to meet it by bringing in more resources, or negotiate the deadline or deliverables with the client well in advance.

- If the client changes their mind about something, use your communication skills to work out the best solution collaboratively. Avoid potential issues related to extra work by including a clause in your contract clearly stating what happens if the project brief changes along the way.
- If you suspect that a senior manager in the client organization is not entirely positive about the project, try to have a conversation about what you're trying to achieve in terms that make sense to him or her. Ask for any objections and concerns, and be able to demonstrate how you're addressing them. Try to ensure that all senior stakeholders have input early on, so they don't have it later by changing everything!
- If the client asks you to do something not outlined in the brief, explain that you can do so but only after the current project is complete. Put the new request on a wish list that you can review with the client later and agree on deadlines and costs.

Problem-Solving Skills

Creating a website for a client is a form of problem-solving. They have a problem (typically, not having as many customers as they'd like), and you are brought in to help them solve it. Any effective website will be an answer to a problem, rather than an end in itself, and the site needs to evolve and grow to continually address that problem throughout its lifetime—a good reason to use WordPress, whose flexibility enables a site to grow after launch.

All the areas we've already discussed involve solving problems—such as dealing with delays or disagreements—but the very process of designing and developing sites, themes, or plugins for clients involves problem-solving skills. If you get into this mindset of viewing your job as solving a problem, you can create solutions that truly benefit your clients, whether they're hiring you to build them a website or buying themes or plugins from you.

Before planning any project, consider what problems it has to address. What are the objectives? What does the client want people to do when they visit the site? What do the visitors expect when they are navigating the site? What does success look like?

This approach applies on both a large scale, when looking at a site's overall objectives, and on a small scale, when deciding exactly how to approach a given part of the site. For example, suppose the client wants to sell something online. Do they need a full-blown e-commerce solution, a subscription-based site, or a simple form linked to PayPal? In order to provide the client with the best solution, you first need to understand the client's problem (goal) and what they need to achieve.

Developing Your WordPress Skills: It Never Stops

You've already learned about some of the ways you can develop your WordPress skills by getting involved in the WordPress community. As someone selling WordPress services to others, be they clients or other WordPress users, you need to remember the following important point: Keep developing—continually. WordPress (like web design and development in general) never stands still—ongoing releases and updates mean you need to keep current with changes to the platform itself. In addition, methods for working with WordPress and extending it are constantly developing, so in order to remain competitive you have to keep up.

Set aside time every day, or at least every week, to learn about WordPress developments or techniques. Some of this time should be spent reading blogs, magazines, and journals to keep abreast of new developments, while some time should be spent expanding the range of WordPress skills in your armory.

Identify any gaps in your skills and the areas you need to improve on in order to win the kind of contracts you want, meet your clients' needs better, expand your services, or simply feel more confident in your own ability. Identify experts in those areas and follow their blogs or articles. Buy, read, and work through the growing library of WordPress books. Follow the blog at http://make.wordpress.org to keep abreast of changes, updates, and issues.

Set aside a budget for your own (and your team's) development. This should include money for books, conferences, and other events. As well as a monetary budget, you'll need a time allocation. When do you learn best—reading RSS feeds and blogs over your cornflakes, poring over a book in the wee small hours, or experimenting with new technologies at a hackday? Identify your own style and then block out some suitable time.

There will inevitably be occasions when personal development has to take a back seat to client work—if your business is successful, you'll struggle to juggle these. Don't let these periods of time last too long. If you get out of the habit of developing your WordPress skills and knowledge, you'll start to lose your edge—and if you lose your expertise, you'll start losing potential clients.

I recommend that every WordPress professional attend at least one WordCamp every year. This is where you'll meet other like-minded developers, pick up new techniques, make contact with potential partners, clients, or staff, and get out from behind your desk and mix with some real people.

Giving Something Back to the WordPress Community

Becoming a WordPress professional isn't just about filthy lucre. Because WordPress is an open-source platform, its continued evolution and success depends on a broad, highly committed community of developers who contribute to WordPress in a variety of ways.

Becoming part of this community helps you develop your WordPress skills, network with other WordPress developers, and raise your profile in the community. It also helps WordPress to evolve so that it continues to meet the needs of users, designers, and developers—including you!

Contributing to the Development of WordPress

WordPress has a team of core developers, but they aren't the only people involved—hundreds of developers contribute code, without whom WordPress couldn't survive. Areas you can contribute to include the following:

- Submitting patches
- Raising tickets
- Contributing to UI and design development
- Translating WordPress
- Reviewing themes

If you are going to be involved in core development, we recommend taking some time to understand what's involved and what's expected. Talk to other WordPress contributors if you can, or ask questions on the Make WordPress site at http://make.wordpress.org. The site also describes how to contribute, including detailed guidelines on submitting patches.

Testing

In addition to needing developers to contribute to WordPress core, WordPress also seeks people to help with testing. This isn't limited to developers. To get started, you need to install the WordPress beta tester plugin and set it up to work with one of two types of build—point release nightlies or bleeding edge nightlies. Point release nightlies tend to be more stable, and are useful for testing your own themes and plugins against beta versions of WordPress, while bleeding edge nightlies represent the very latest code and are what you would use as a WordPress tester.

You can also report bugs you find while testing beta versions of WordPress or already released versions.

Writing Free Plugins and Themes

Writing code for release to the public is a great way to develop your WordPress skills, even if you've never written a plugin or theme in your life. By making your code available free on the WordPress repository, you are helping other developers and giving them the opportunity to contribute to and improve your code, which helps you learn.

If you're a theme developer, why not try releasing a free plugin? Similarly, plugin developers can extend their knowledge by creating a theme. Other developers will help you to improve upon what you've written, and it gives you a chance to work with WordPress from a new angle.

This book has a whole chapter dedicated to releasing your code to the public (both free and paid). If this is something you want to do, take a look at Chapter 12.

Contributing to the WordPress Website

The content of the WordPress site at `http://wordpress.org` is contributed by developers just like you, and if you're good at answering questions or decsribing how code works, you can help.

Contributing to the Codex

The Codex (`http://codex.wordpress.org`) is the place most WordPress users and developers go to for information on how WordPress works. You can volunteer to contribute to the WordPress Codex, helping to improve the quality of the documentation. As WordPress evolves, there will always be functions and features that need documenting, or you could improve, add to, or update existing documentation. To find out how to get involved, see `http://codex.wordpress.org/Codex:Contributing`.

WordPress Support Forums

Open-source communities depend on the active participation of their members. If you know the answer to a question that another user has posted on a forum, please add your reply; it could help many other developers. You can use tags to access forum threads on a specific topic that you have expertise in—for example, if you're a CSS developer, visit `http://wordpress.org/tags/css` to find questions you might be able to answer.

Getting Involved in WordPress Groups or WordCamps

Local WordPress Groups and WordCamps are an invaluable source of support, informaiotn and advice for WordPress users and developers. Around the world, people meet up to talk about WordPress, to hear about techniques from speakers, and to discuss the work they're doing with WordPress—as well as to make friends!

Helping with Organization

If you've got experience of organizing events, or you'd like to have a go, you can volunteer to help organize your local WordPress group of the WordCamp in your country or region. Taking on this role will introduce you to a broad community of other WordPress users and developers, and you will gain a lot of respect and gratitude from your peers. But most importantly, it could mean an event can take place in your locality that otherwise wouldn't—which will benefit many WordPress developers and users.

You can find out more about becoming a WordCamp organizer at `http://central.wordcamp.org/become-an-organizer/`.

Contributing as a Speaker

If you have WordPress knowledge or experience that others could find useful and you don't mind talking in front of an audience, speaking at a WordCamp is a great way to share your expertise, learn from other participants and speakers, and get some public speaking experience in front of a generally friendly audience. I've done this and gained a lot from it.

How much and exactly what you're able to take on will depend on your experience, areas of expertise, and availability, but getting involved in the WordPress community is a great thing to do if you are truly serious about using the platform. By being connected in this way, you'll learn about the latest developments in WordPress itself, you'll meet people from whom you can learn a huge amount and with whom you may be able to work in the future; and you'll be introduced to a potential source of WordPress developers you can later hire when you need more resources.

The Potential for Professional WordPress Development

At this point, we've already established that WordPress isn't just for bloggers anymore; it's a fully fledged, grown-up web development tool being used across the board—from the smallest of startups to major multinational corporations.

As a WordPress developer, however, you need to understand just how you can harness WordPress to meet your clients' needs and help them build their business, as well as your own. This is about more than simply building custom WordPress sites, or developing themes or plugins. It's about understanding the benefits that WordPress can bring to your clients, and being able to communicate this in language that makes sense to them—in plain English.

It's also about recognizing opportunities and gaps in the market, particularly if you're developing themes or plugins—if you fill a niche, you'll be much more successful than if you produce yet another Twitter plugin.

Harnessing WordPress As a CMS

At some point, a client will utter those dreaded words:

> *"WordPress – but I thought that was just for blogs?"*

Resist the temptation to groan or roll your eyes in mockery. Unfortunately, WordPress is still perceived as a tool used exclusively by mommy bloggers and geeks to fill the Internet with personal impressions and ideas (not that we have anything against anyone doing that!). In order to sell WordPress to your clients, you need to be ready to explain that WordPress is much more than that. In particular, you need to develop arguments supporting the use of WordPress as a CMS—indeed, that it's in fact better than any custom CMS another agency may try to sell them.

The obvious arguments are about WordPress's development and credibility:

- WordPress is used by major organizations with huge, complex sites containing data they absolutely don't want to lose, such as CNN, the Daily Telegraph, and more. For some great examples, see the showcase page on the WordPress site at `http://wordpress.org/showcase/`.
- WordPress is much more secure than many people assume, and it can be made more secure. It is used by organizations and public entities that go to great lengths to avoid being hacked—1600 Pennsylvania Avenue and 10 Downing Street, anyone?
- As a tool used by thousands of developers, WordPress is a platform your client can stick with even if you stop working with them. Don't be afraid to tell your clients this; we believe that reassuring your clients that their site can be developed by another developer or agency in the future enhances their trust in both WordPress and you. If you have ever had a client who was switching from their old agency's custom CMS and could no longer update their site, you'll know how much of a problem this can be.

As well as the arguments in favor of WordPress, you should also be prepared to explain your ability to modify WordPress for your client's specific scenario, effectively turning it into a custom CMS but without the drawbacks:

- You can configure the WordPress installation to increase its security and avoid any potential problems related to hacking or server problems.
- You can structure the site content in any way the client needs using custom post types, taxonomies, and fields, resulting in a unique site that is very different from standard WordPress.
- You can design and code a custom layout and design that is in harmony with the client's branding, it can be as innovative as necessary, and it doesn't need to have a traditional blog layout unless they want that.
- You can add plugins, scripts, and more to the site to give it extra functionality, creating a better user experience—changing the site from one that merely enables users to consume content to one that encourages them to interact with the site in new ways.
- You can customize the WordPress admin to make editing and managing the site quick and easy for the client, to point them in the direction of the relevant content and to reflect their brand if necessary.
- You can configure the client's site to be lean and fast to load, stripping out unwanted functionality and using caching and other techniques to improve performance.

- You can install and configure SEO plugins (and teach your client to do so), which will enhance the site's search engine rankings.
- You can create a responsive site or web app for your client, providing them with another way to reach and potentially sell to their customers.

Of course, you may not be able to do all these things—yet. Nonetheless, they represent a cross-section of the ways in which you can push, stretch, and customize WordPress—and you will be able to do them after you've finished this book!

WordPress for SEO and Site Conversions

Developing a fabulous site won't help your client much if people can't find it. This is obviously essential for a business—attracting the right customers to a site will increase the amount of sales. But even sites that don't sell anything or offer a service, such as community sites, need to attract the right visitors in order to be successful, as their purpose is to spread information or affect public opinion.

Some clients will have their own techniques for pulling visitors to their site, but the majority will be reliant on search engines, via either organic search (i.e., free) or fee-based search listings such as Google AdWords.

The great news is that WordPress gives you and your clients a head start in Search Engine Optmization(SEO). This isn't just about the use of plugins:

- WordPress (when paired with a well-written theme) uses valid, standards-compliant code, which is a fundamental starting point for SEO as well as for good web development.
- If you write your theme in HTML5 (which you should), you'll have access to semantic elements that help search engines index your code.
- WordPress can be configured to be fast, using off-the-shelf plugins and your own customization.
- WordPress automatically generates search-engine-friendly page titles based on your site title and page or post title.
- Permalinks can be configured to enhance URLs within the site for an SEO boost.
- The image uploader makes adding `<alt>` tags and titles for images easy, which enhances not only SEO but also accessibility.
- By installing a plugin such as SEO by Yoast, you or your client can specify titles and descriptions site-wide or for individual pages or posts, and analyze how well a page is likely to perform.
- Other plugins let you link a Google analytics account to the site, displaying statistics in the dashboard that clients can view every time they work on the site.

Developing Themes and Plugins You Can Sell

Using WordPress to build sites for clients is just one way to make a living from the platform. You could also develop plugins or themes that you can then sell to other WordPress users.

This isn't as easy as you might think. A lot of people are reluctant to part with their hard-earned cash for themes, frameworks, or plugins when so many are available free. If you hope to make any sales, you have to offer something

that is of very high quality, gives developers or users something they really need, and fills a need not currently being met by another product. You may have a great idea for a Twitter plugin, for example, but with so many already available, is it really a good idea to put all your efforts into competing in such an overcrowded market?

To make a success of this, you need to do your R&D first:

- Research the market. Does your idea fill a gap or provide something that is of considerably higher quality than what's already out there?
- Talk to developers and users, and find out what they want from a plugin or theme like yours. Identify influential WordPress bloggers and experts and ask them for their views. If they like what they see they may recommend it to their followers.
- If you are developing a theme, research the avenues for selling them. The easiest way is via a large theme vendor. If you go that route, find out what percentage of your sales they take and the price of similar themes.
- If you're developing a plugin, you'll have to find your own ways of marketing it and making it available. You might consider releasing a free version on the WordPress plugin repository, with a premium version for people who want extra features. Take care to make your free version useful to a respectable proportion of your audience, however, or you may inspire frustration instead of a loyal following.
- If you're developing a framework, you need to familiarize yourself with those already available. Until you understand what they do and how they work, you can't offer something new to fulfill a different need.

You also need to think about the way you build your product:

- It nearly goes without saying that your code must be reliable, bug free, compatible with the latest release of WordPress (and ideally the previous release too), accessible, and standards compliant.
- Consider the number of options you'll make available via the WordPress admin. Buyers of premium themes and plugins expect to be able to make their own customizations. You can also provide developers with documentation on customizing the code.
- If you're building a premium product, other developers won't help you improve upon and develop it (unlike a free one). You'll need to budget for the time or expense of doing this, and make updates available when new versions of WordPress are released.
- Ensure that your plugin or theme is robust and not liable to hacking—you don't want to be responsible for hundreds of WordPress sites going down because of a vulnerability in your code!

There's a lot more to think about and take into consideration when developing themes, frameworks, or plugins for release or sale. For more details, see Chapter 12, "Releasing Your Code to the Public."

Summary

Hopefully I've convinced any skeptics out there that WordPress is a great platform on which to build a business. It's evolved way beyond its humble beginnings as a blogging platform, and now offers the capable developer myriad ways to extend, push, and customize it to power large, complex sites that would be hardly recognizable as a typical WordPress site.

As a web development professional, you can use WordPress to develop websites for a range of clients, as well as release and maybe sell products that will help other users and developers to extend and work with WordPress themselves, in the form of plugins and themes.

WordPress is at a hugely exciting stage in its development, and it's rapidly becoming the mainstream CMS of choice for clients, developers, and users. If you're ready to take advantage of WordPress, there's a lot you need to learn, so read on…

Further Resources

Professional WordPress Development

"How to become a top WordPress developer"
`http://wp.smashingmagazine.com/2012/08/23/how-to-become-a-top-wordpress-developer/`

"Do's and don'ts for WordPress startups"
`http://wp.smashingmagazine.com/2012/06/21/dos-donts-wordpress-startups/`

Client Centric Web Design by Paul Boag (Boagworld)
`http://amzn.com/B007QUTLQ6`

How to Start Your Own Business for Entrepreneurs by Robert Ashton (Pearson Business, 2012)
`http://www.amazon.co.uk/dp/0273772171`

Coding Practices

W3C validator
`http://validator.w3.org`

Web standards project
`http://webstandards.org`

Web accessibility standards
`http://www.w3.org/WAI/`

WordPress coding standards
`http://codex.wordpress.org/WordPress_Coding_Standards`

WordPress guidance on validating your site
`http://codex.wordpress.org/Validating_a_Website`

Designing with Web Standards by Jeffrey Zeldman and Ethan Marcotte (Voices That Matter)
`http://amazon.com/0735712018`

"Top 15 Best Practices for Writing Super Readable Code"
`http://net.tutsplus.com/tutorials/html-css-techniques/top-15-best-practices-for-writing-super-readable-code/`

Giving Something Back to the WordPress Community

Contributing to WordPress
`http://wordpress.org/Contributing_to_WordPress`
`http://make.wordpress.org`

Aaron Campbell and Andy Stratton on getting involved (video)
`http://wordpress.tv/2011/09/08/aaron-campbell-and-andy-stratton-getting-involved-contribution-and-courtesy/`

WordPress Core Contributor Handbook
`http://make.wordpress.org/core/handbook/`

Guide to WordPress beta testing
`http://make.wordpress.org/core/handbook/beta-testing/`

WordPress beta tester plugin
`http://wordpress.org/extend/plugins/wordpress-beta-tester/`

WordCamps
`http://central.wordcamp.org`

WordPress meetups
`http://wordpress.meetup.com`

The WordPress Codex
`http://codex.wordpress.org`

WordPress support forums
`http://wordpress.org/support/`

WordPress trac
`http://core.trac.wordpress.org`

WordPress support forums
`http://wordpress.org/support`

Make WordPress site
`http://make.wordpress.org`

Developing Your WordPress Skills

WordPress TV: Videos from WordCamps and other events
`http://wordpress.tv`

WordPress blog
`http://wordpress.org/news/`

The WordPress Codex
`http://codex.wordpress.org`

Smashing Magazine's WordPress section
`http://wp.smashingmagazine.com`

Roundup of professional WordPress development tools
`http://wp.tutsplus.com/articles/general/professional-wordpress-development-tools-2/`

WordPress for SEO

All in One SEO Pack plugin
`http://wordpress.org/extend/plugins/all-in-one-seo-pack`

SEO by Yoast plugin
`http://wordpress.org/extend/plugins/wordpress-seo`

WordPress SEO tutorial
`http://yoast.com/articles/wordpress-seo`

Developing Themes and Plugins You Can Sell

Professional WordPress Plugin Development by Brad Williams, Ozh Richard, and Justin Tadlock (Wrox, 2011)
`http://amazon.com/0470916222`

WordPress Plugin Developer Center
`http://wordpress.org/extend/plugins/about/`

Theme review guidelines
`http://codex.wordpress.org/Theme_Review`