SPONSORED BY

GeoTrust®

GEEK GUIDE

DIY Commerce Site

LINUX™ JOURNAL

Table of Contents

REUVEN M. LERNER is a Web developer, consultant, trainer and longtime columnist for *Linux Journal*. He recently completed his PhD in Learning Sciences from Northwestern University. You can read his blog, Twitter feed and newsletter at http://lerner.co.il. Reuven lives with his wife and three children in Modi'in, Israel.

GEEK GUIDES:
Mission-critical information for the most technical people on the planet.

Copyright Statement
© 2015 *Linux Journal*. All rights reserved.

This site/publication contains materials that have been created, developed or commissioned by, and published with the permission of, *Linux Journal* (the "Materials"), and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of *Linux Journal* or its Web site sponsors. In no event shall *Linux Journal* or its sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

No part of the Materials (including but not limited to the text, images, audio and/or video) may be copied, reproduced, republished, uploaded, posted, transmitted or distributed in any way, in whole or in part, except as permitted under Sections 107 & 108 of the 1976 United States Copyright Act, without the express written consent of the publisher. One copy may be downloaded for your personal, noncommercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Linux Journal and the *Linux Journal* logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners. If you have any questions about these terms, or if you would like information about licensing materials from *Linux Journal*, please contact us via e-mail at info@linuxjournal.com.

About the Sponsor

GeoTrust—A Trusted Leader in Online Security Services

GeoTrust is the world's second largest digital certificate provider. More than 100,000 customers in over 170 countries trust GeoTrust to secure their websites and online transactions. Our range of digital certificates enable organizations of all sizes to maximize the security of their websites and digital transactions cost-effectively.

GeoTrust's uncompromised world-class SSL Certificates (https://www.geotrust.com/ssl/) offer fast delivery at a cost-effective price, enabling up to 256-bit SSL encryption, and include the GeoTrust Secured Seal, which is generated in real time by GeoTrust servers to show customers that the site is currently protected.

The GeoTrust Security Center, a robust management portal, makes the task of managing these certificates intuitive and simple. GeoTrust also offers volume pricing for Enterprise SSL customers that need 10 or more certificates (https://www.geotrust.com/enterprise-ssl-certificates/enterprise-ssl).

DIY Commerce Site

REUVEN M. LERNER

Introduction

If you have been living in a cave for the last 15 years, I have some exciting news for you: the Internet is hopping with e-commerce opportunities. If you have a product and you're willing to put in the time to market it, you can sell that product on-line. If your product is good and you market it well, you even can make a lot of money.

For those of you who haven't been living in a cave for some time and have been using the Internet in your personal and professional lives, the notion of making money from e-commerce may seem like something you'll eventually get around to doing in the future, or that you

might do if it weren't a pain. After all, starting to sell on-line must be difficult, right?

The answer, as you'll see in this ebook, is a resounding "No." Creating your own e-commerce site and starting to sell your own products and services on-line is much easier than you might think. Indeed, the technological parts of an e-commerce site almost certainly are going to be the easiest parts of making money on-line. It'll take much more time (and be much more complicated) to decide what you're going to sell, create the product or service and market it. Indeed, you should expect that the bulk of your time as a budding e-tycoon will be spent on marketing, rather than on the creative and technical side of things.

I've been working with e-commerce for nearly 20 years as a consultant and Web developer, dealing with dozens of clients who have e-commerce needs. I've experienced the thrill of seeing businesses bring in large amounts of money from customers they've never met face to face, buying and selling goods and services on-line.

More recently, I've started to self-publish ebooks, which has provided me with an even greater thrill. Indeed, it's amazing to wake up in the morning and receive notifications from your e-commerce system indicating that you have sold copies of your product to people you don't know. Actually, that's not true. Even better than making the sale to someone you don't know is receiving e-mail from people several days later, telling you that your book has provided them with useful information, and that they appreciate it.

In this Geek Guide, I walk through the process of creating

an e-commerce shop where you can sell your own products and services. By the time you're done reading this, you will be ready to put together a simple Web site and start selling.

What Can You Sell?

The first question to consider is what you can and will sell on-line.

Digital Goods: This is probably the easiest way to start with e-commerce. "Digital goods" often refers to ebooks, but it also can mean videos, audio recordings, or even software or music. Creating and selling digital goods requires you to create a file you want to share with others. In most cases, that means just following your usual procedures for creating a PDF file or any other format you wish to use. However, many digital goods publishers provide a number of formats; for example, I have published ebooks in .pdf, .epub and .mobi formats to accommodate customers who use different e-book readers. Purchasers of my higher-cost packages also get videos. This means I need to make the videos available either for download or streaming.

Services: The SaaS (Software as a Service) model for on-line commerce is extremely popular and provides an excellent business model for a variety of services. Famous examples of this are Basecamp and GitHub, but cloud server providers, such as Amazon and Rackspace, also use this "pay for what you use" model. I subscribe to a variety of such services for my e-mail, hosting, e-mail list hosting, user analytics and server performance monitoring. There are SaaS providers for a wide variety of services nowadays, and

you certainly can join the party—providing a full-fledged application aimed at end users or a software service aimed more at developers.

Because it potentially can bring in a great deal of money, people often think of SaaS as the first step in their e-commerce careers. However, many experienced hands in this area have indicated, and rightly so, that starting with a small, simple digital good you sell at a relatively low price is probably the best way to dip your toes into e-commerce waters. It requires less software development, allows you to explore issues of marketing and customer acquisition, and if things fail, they'll at least fail with a minimum of investment. SaaS generally requires so much time and money just to test the market that you probably should hold off on considering it until you have a bit of experience under your belt.

I should note that other types of services can be sold on-line, such as Webinars and consulting engagements. I don't address those types of services directly in this guide, concentrating more on the idea of selling products and specifically digital products. However, many of the considerations regarding payment discussed here will be appropriate even for service operations.

Physical Goods: Physical goods are both the easiest and the hardest things to sell on-line. On the one hand, selling physical goods is something everyone understands. I have a widget. You want the widget and pay me for the widget, and I ship the widget to you. If you've watched *Shark Tank* on TV or perused sites like eBay and AliExpress, you know the number and variety of widgets

The idea is that when customers buy the product, they receive—via download, e-mail attachment or link sent via e-mail—the files that you created.

people sell on-line is huge.

At the same time, you'll also realize the need to manufacture and ship the products, along with all of the other tasks. Thus, although this Geek Guide might provide some information on how to set up an e-commerce site for your physical goods, it concentrates on digital goods, which don't require either manufacturing or shipping.

You Have a Digital Product. Now What?

Congratulations! You have created a digital product, which might be anything from a $5 quick-reference guide to a $5,000 pre-recorded on-line course. From the perspective of setting up an e-commerce shop, it really doesn't matter. A "product" may contain, as indicated above, any number of files. The idea is that when customers buy the product, they receive—via download, e-mail attachment or link sent via e-mail—the files that you created.

So, how do you go about letting people purchase your product?

In many ways, the cheapest and easiest way to go, especially if you have a small number of digital products, is not to set up your own server. That's the choice I've

made with my own ebooks, simply because the overhead, administration and maintenance of my own e-commerce server couldn't be justified. I decided to outsource my on-line shop—something you definitely should consider.

Outsourcing an on-line shop makes life much easier and removes the need to deal with e-commerce nitty-gritty, such as taking credit cards. However, it removes a degree of control, and it also means you'll be paying a higher set of fees than if you were handling things yourself.

Two outsourcing firms I have used with my digital products are DPD (http://getdpd.com) and Gumroad (http://gumroad.com). The pricing and service models, as well as user interfaces, are different, but the basic idea is the same. You create a new digital product and give it a name, price and description. You then upload one or more files to the company's servers. Once you've finished uploading the files, you indicate that your product is complete, and you're in business.

Each of these providers has its own advantages. I moved from DPD to Gumroad in the past few months and generally have been quite happy with the results. Among the reasons I switched were Gumroad's user interface (which provides JavaScript-based pop-ups that you can embed on another page) and the ability to stream videos to users rather than require users to download them.

An important consideration is when (and how) you get paid by a third-party vendor. In the case of both DPD and Gumroad, payments are sent to you via PayPal. DPD pays each time you make a sale, and Gumroad pays once every two weeks for all sales executed since the last payment.

This means if you don't want to use PayPal to receive payments and would prefer another option, such as ACH or IBAN transactions, you might be out of luck.

DPD and Gumroad have slightly different ways of charging. In the case of DPD, you pay a monthly fee for a certain amount of disk storage for your files, plus a percentage of every sale. Gumroad takes a larger amount (percentage + flat amount) per sale, but has no monthly fees or limits on file uploads.

Another option to consider, especially if you have a large number of products, is Shopify—an SaaS service designed to help create on-line stores. Shopify is designed for all sorts of stores, including those that ship physical goods as well as digital products. I haven't used Shopify, but I have heard good things about it. Nevertheless, given that I have a small number of digital products and my own Web site, I've been content using Gumroad.

The Nitty-Gritty of E-commerce

When you use your credit card to buy something at a store, the process seems instantaneous and straightforward. However, in order to create an e-commerce site, you need to understand much more of what's happening behind the scenes.

Let's say you use your credit card to buy something that costs $100. The store doesn't receive the full $100. Rather, the credit-card company takes a percentage of the sale. How much the credit-card company takes depends on the card and the store's agreement with the company. It might be 2.5%, or even 5% or 6%, depending on what

was negotiated, based on store history, expected sales and potential fraud. If there is a great deal of fraud at your store, you might need to pay more in the future. By contrast, if you're a good and reliable customer, the credit-card companies will reward you with a lower percentage and/or per-transaction fee.

You might think that every time someone purchases something from your store, the credit-card company will send you a payment. This isn't the case. Just as you pay your credit-card bill once per month, you'll also get paid by the credit-card company into your "merchant account", which you can think of as a credit-card account in reverse.

So your e-commerce store will need to communicate with the credit-card company every time there is a sale, right? Well, not exactly. Your business is too small to speak directly with the card company. Instead, you'll need to go through a payment gateway service, which talks to the credit-card company on your behalf and (of course) takes a cut as well.

This means before you can start selling, you'll need to have a merchant account and choose a gateway, which can be a frustrating process. It's no surprise that companies like PayPal, Stripe and Gumroad have done so well—they allow you to deal with a single company for your e-commerce needs rather than two companies, bureaucracies and paperwork. DPD doesn't try to abstract that away; it assumes that you have a merchant account and gateway or that you'll use PayPal if you don't want to deal with the headaches involved with such systems.

Some companies have tried to simplify this process by providing multiple services under one roof. One of the most famous, Braintree Payments, was well known for its developer-friendly service and APIs, and for offering both gateway and merchant-account services. Braintree was acquired by PayPal in 2013, but it continues to operate such services for a wide variety of companies. My experience with Braintree has been quite positive, especially when it comes to customer service. If you need a gateway and/or merchant account, I'd certainly suggest speaking with Braintree.

Unfortunately, things are even more complicated if you're located outside the United States, because many US-based payment operators aren't willing to work with foreign companies. When I started selling products on-line via DPD, the only option I had for a gateway or merchant account was PayPal. This didn't affect my customers, who could come from anywhere, but it did limit my ability to compare different options and choose the lowest-cost combination. Gumroad, to its credit, works with vendors from around the world and accepts payment via both credit cards and PayPal.

Taxes

One of the most difficult aspects of handling purchases is taxes, particularly sales taxes. In the US, there are multiple, often overlapping, taxes that need to be collected. Depending on where you are located and where your customer is located, you might need to charge state, county and/or city sales taxes. If you and/or your customers are located in Europe, a different set of

taxes might apply.

Keeping track of this is extremely difficult, and you're almost certainly never going to get it right on your own. And of course, tax rates can and do change over time, as governments raise and lower them for particular regions and on particular products. For this reason, you should make sure that any e-commerce solution you choose handles taxes automatically.

DIY E-commerce

Let's assume you have decided, for whatever reasons, to go it alone—you don't want to use a third-party sales system. Rather, you want to create your own e-commerce server and sell digital goods via that system. The easiest and fastest way to do so is likely WooCommerce, an open-source system based on the well known open-source WordPress platform.

Other systems exist, but WooCommerce is easy to install, flexible and widely used, claiming 30% of all on-line stores. This means if you get stuck, you'll likely be able to find support. It also means if there are security problems or other bugs, they likely will be fixed quickly. WooCommerce is distributed under the GNU Public License (GPL) and is developed by Automattic, the same people as WordPress.

For the purposes of this Geek Guide, let's create a new WooCommerce-based store from scratch. First, create a tiny new virtual Ubuntu server at Digital Ocean (I chose it for this example mostly because of the speed, low cost and convenience it offers). After creating your "Droplet" server,

log in to the computer as the root user, using the password supplied by Digital Ocean (which you subsequently should change) and update the system:

```
# apt-get update
# apt-get upgrade
```

Once the upgrades are fully in place, install Apache, along with its PHP module:

```
# apt-get install apache2 libapache2-mod-php5
```

If you prefer a different HTTP server, such as nginx, you may well want to use it (I still prefer Apache, although it has been demonstrated that nginx scales better under heavy loads):

```
# apt-get install mysql-server
➥libapache2-mod-auth-mysql php5-mysql
```

When installing MySQL using `apt-get`, you'll be asked to enter a "MySQL root" password. This is not for the Linux-level root user, but rather for the MySQL administrator, inside the database, so the password should be something different from your root password.

Now, it's possible to install WordPress via `apt-get`, but I (like many other Web developers) prefer to install things on my own, such that I'm operating independently of the Ubuntu packaging system. This means setting up your own server for WordPress. Fortunately, that's extremely easy to do.

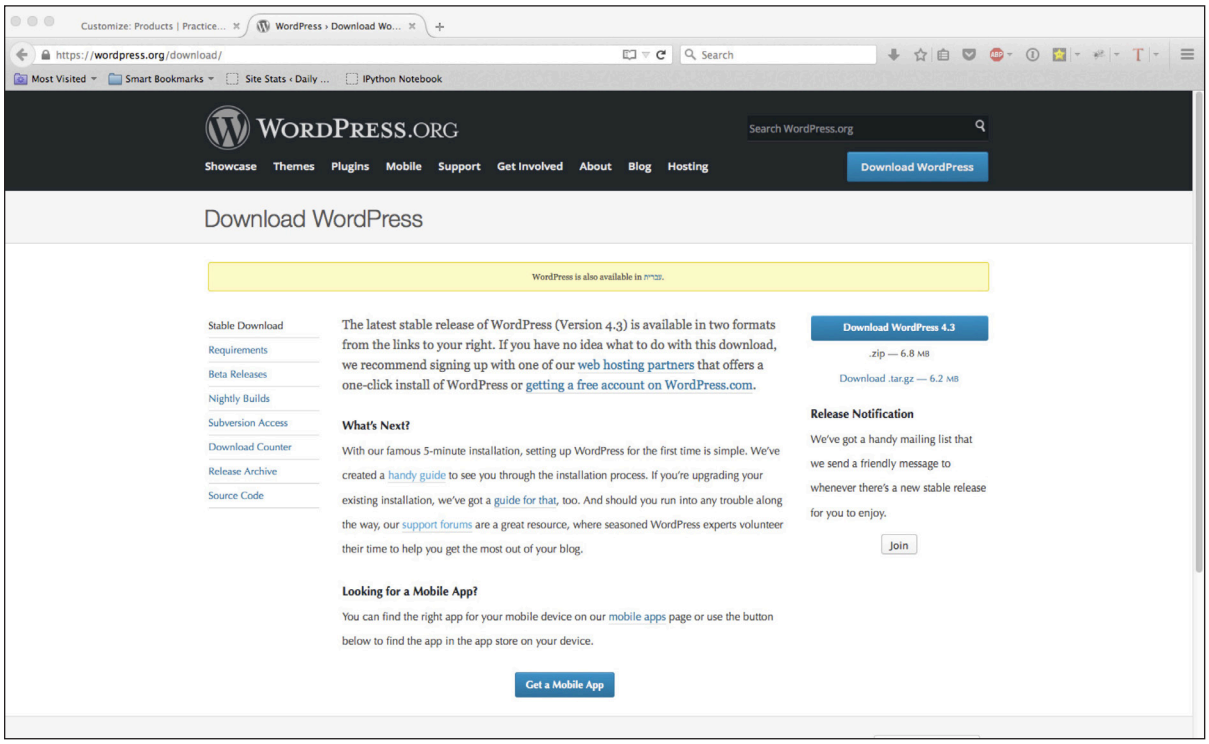

FIGURE 1. Downloading WordPress

First, download the WordPress software. Because this example uses Ubuntu, the assumption is that all Web-related items go into /var/www, so execute the following:

```
# cd /var/www
# wget http://wordpress.org/latest.tar.gz
# tar -zxvf latest.tar.gz
```

When this is done, you'll have a "wordpress" directory under /var/www. Because Apache runs under the "www-data" user in Ubuntu, you then change the ownership of all WordPress-related files and directories

to be "www-data":

```
# chown -Rv www-data wordpress
```

In the above command, `-R` stands for "recursive", meaning it'll affect every subdirectory and file. The `-v` option means verbose, an option I often like to use, if only because it guarantees that I'll see what is happening as my command executes.

With WordPress in place, it's time to tell Apache where the WordPress installation is. Modern versions of Apache allow you to have multiple Web sites, each of which with its own configuration file. On an Ubuntu system, sites that you can potentially be running have their configuration files in /etc/apache2/sites-available. When you want to enable a site, you make a symbolic link to the appropriate file in "sites-available" from the "sites-enabled" directory.

For this example case, however, let's say you don't want to have multiple sites—just one site. For that reason, modify the configuration of the default Apache site, whose configuration file is typically in /etc/apache2/sites-enabled/000-default.conf. With comments removed, the file looks like this:

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

In order to make Apache work with WordPress, you'll need to modify the above a bit. The most important thing to do is modify the `DocumentRoot` directive to point to the WordPress configuration you've just added. You'll also indicate that "index.php" should be used as the default page when none is specified. Finally, you'll add an e-mail address that actually works for people who want to contact you. The resulting configuration file looks like this:

```
<VirtualHost *:80>
    ServerAdmin reuven@lerner.co.il

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    DocumentRoot /var/www/wordpress
    DirectoryIndex index.php
</VirtualHost>
```

Under Ubuntu, you can restart Apache with the command:

```
# service apache2 restart
```

You likely will get a warning message indicating that the `ServerName` directive hasn't been set. This isn't strictly necessary, but if you want to, you can add a line like the following to /var/www/apache2.conf:

```
ServerName  YourServerNameHere.com
```

Restart the server once the name has been set there, and the warnings should disappear.

This is sufficient from Apache's perspective, but it's not quite enough to have WordPress running. That's because you'll need to create a MySQL database in which your WordPress (and WooCommerce) system can run. The easiest way to do this is to enter the MySQL system (from the command line), create the database and set the appropriate permissions:

```
# mysql -u root -p
```

Here, you'll need to enter the MySQL "root" password you set earlier, when you installed MySQL. Once in MySQL, you'll see the `mysql>` prompt. Here, you'll create the database and set the privileges:

```
mysql> GRANT ALL PRIVILEGES ON wordpress.*
 ➥TO "wordpress"@"localhost" identified by "password";
```

In the above command, note that the database name is "wordpress", the user name is "wordpress", the hostname is "localhost", and the password is "password". All four of those parameters must match precisely what you're going to enter next into the WordPress configuration screen. Finally, tell MySQL to reload its configuration with the command:

```
mysql> FLUSH PRIVILEGES;
```

You can test the connection from the command line:

```
# mysql -u wordpress -h localhost wordpress -p
```

If you get a `mysql>` prompt, you're probably doing just fine. Now go to your WordPress installation at the root of your domain. (In my case, I'm setting up a site for my new ebook about regular expressions, so I go to http://PracticeMakesRegexp.com, and I get a WordPress configuration screen.) You'll first be asked to choose a language and then to enter the database information that you entered above—the user name, hostname, database name and password. If they match correctly, you'll be asked for some more information to install WordPress, such as the name of the site and your user name. Enter those, plus a password, and after just a few seconds, your WordPress installation will be ready.

WooCommerce

Now that you have installed WordPress, it's time to install the WooCommerce plugin. Like all WordPress plugins, WooCommerce is available via a Web-based installation system. Click on "add new" from within your WordPress installation, and then search for "woocommerce". Click the install button, and now you will have WooCommerce on your WordPress site. This process should take only a few seconds, even in the slowest of circumstances. Activate the plugin by clicking on the appropriate link, and you'll see a new WooCommerce menu on the left side of the WordPress menu bar with a number of sub-menus. Next, you'll be

asked if you want to install static files for WooCommerce; click the button to install them now.

Once WooCommerce is installed, the first and most important place to go is the "settings" page. The first of these, under the General tab, lets you indicate some basic pieces of information about your on-line store. The first questions you'll be asked are about your store's location, where you're authorized to sell and what assumptions (if any) you can make about customer addresses. This is done, in part, so that the WooCommerce system can handle the tax calculations, ensuring that the appropriate amounts are added to the price.

You'll similarly need to choose the currency in which items are sold in your store, as well as how to format numbers and the currency symbol.

Finally, you'll need to click "save changes" before switching to another tab. The second tab is where you can enter the products to the store.

WooCommerce handles a wide variety of types of goods, including shippable goods. When you go to the Products tab, you'll see four links, each of which allows you to configure different aspects of the products you offer:

- General: general configuration, including whether you want metric or English measurements and whether you want to allow for product ratings.
- Display: how products should be shown in your on-line shop.
- Inventory: what do you currently have on hand? If

you want, you can manage your current stock using WooCommerce; it'll even allow you to configure notifications regarding low inventory and what to display for users.

- Downloadable products: if you're creating digital products, this is the most important tab for you. Most likely, you can just keep the defaults, which indicate how files are downloaded, who has access to them and when access is granted (that is, after payment is made).

Although WooCommerce automatically calculates taxes for you, you do need to indicate whether prices include taxes and how the taxes should be calculated. This is potentially a thorny problem, and it's probably something about which you should consult an attorney or accountant to find out what needs to be paid and on what basis. The Tax tab makes it easy to configure those things, but only once you know how you want it to be configured.

The Checkout tab is where you tie in a payment gateway. Four payment gateways already are configured with WooCommerce when you download it—for a direct bank transfer, for receiving payment by check, for cash and PayPal. Thus, if you want to get started selling on-line right away with PayPal, and already have a PayPal account, you can do so without any additional plugins or configurations. This is the simplest and fastest way to get your store started.

If you're interested in using an alternative gateway or service, you can go to the "add-ons" screen for

The fact that WooCommerce is used in so many places ensures that there will be support for nearly any kind of payment solution you want.

WooCommerce, install the plugin for the gateway of your choice and configure it to use the merchant account for which the gateway provides you service. The fact that WooCommerce is used in so many places ensures that there will be support for nearly any kind of payment solution you want. To activate and use a payment solution, click on the "settings" button, and enter your identifying information. Each payment system has its own features; for example, PayPal allows you to process refunds via WooCommerce, assuming that you enter some additional credentials.

Adding a Product

Along the left side of your WordPress administration page, just below the "WooCommerce" configuration menu, you'll find the "Products" menu. This is where, not surprisingly, you add new products as well as new product categories. Although I'm going to have a single ebook on this site, you can imagine that I'll eventually want to have a larger number of products, in a number of categories that people can search for and buy. Thus, I'm first going to go to the "Categories" menu option and set up several categories.

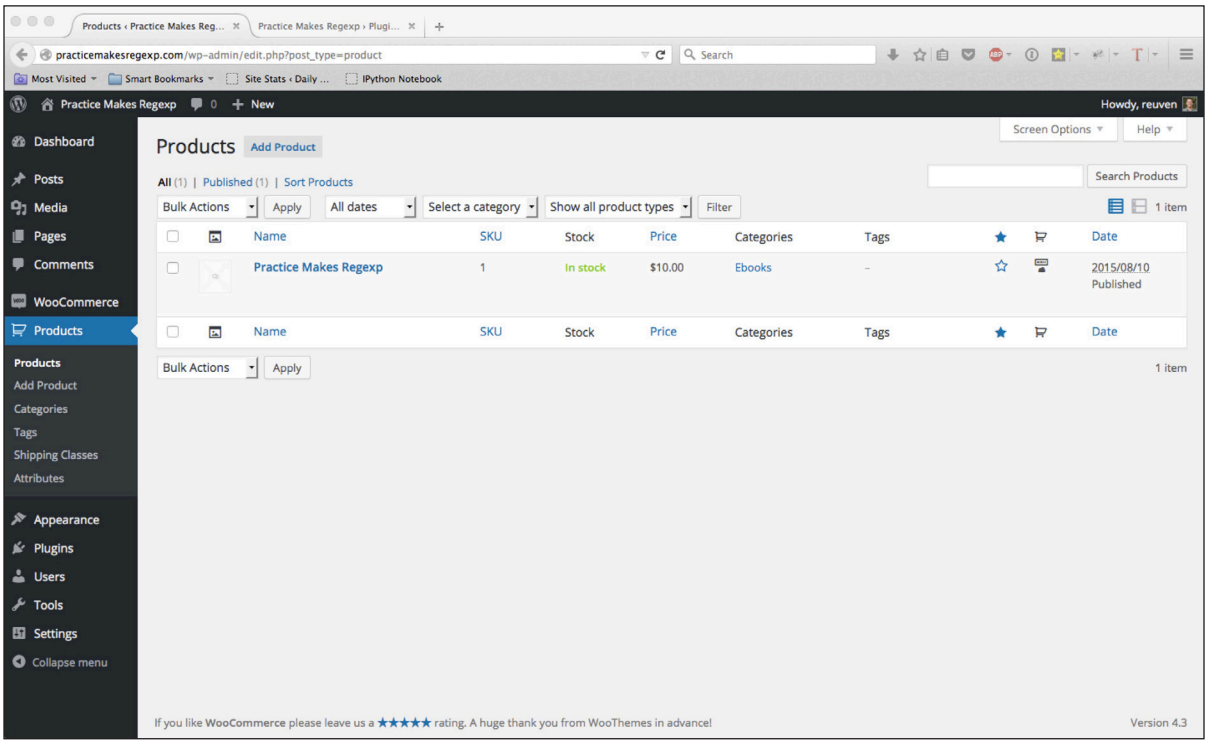

FIGURE 2. **Adding Products to the System**

When you enter this page, you'll automatically be asked for a new product category. Each category has a name that is displayed to users, a "slug" that is contained in the URL and helps to identify your product uniquely, an optional parent (letting you create subcategories) and an optional category description. Your category also may have an image to identify it to potential customers.

Next is the final step of setting up an e-commerce solution—adding a product. If you have any experience with WordPress as a blogging platform, much of this will seem extremely familiar to you. Instead of a posting title and body, you have a product name and description. Instead

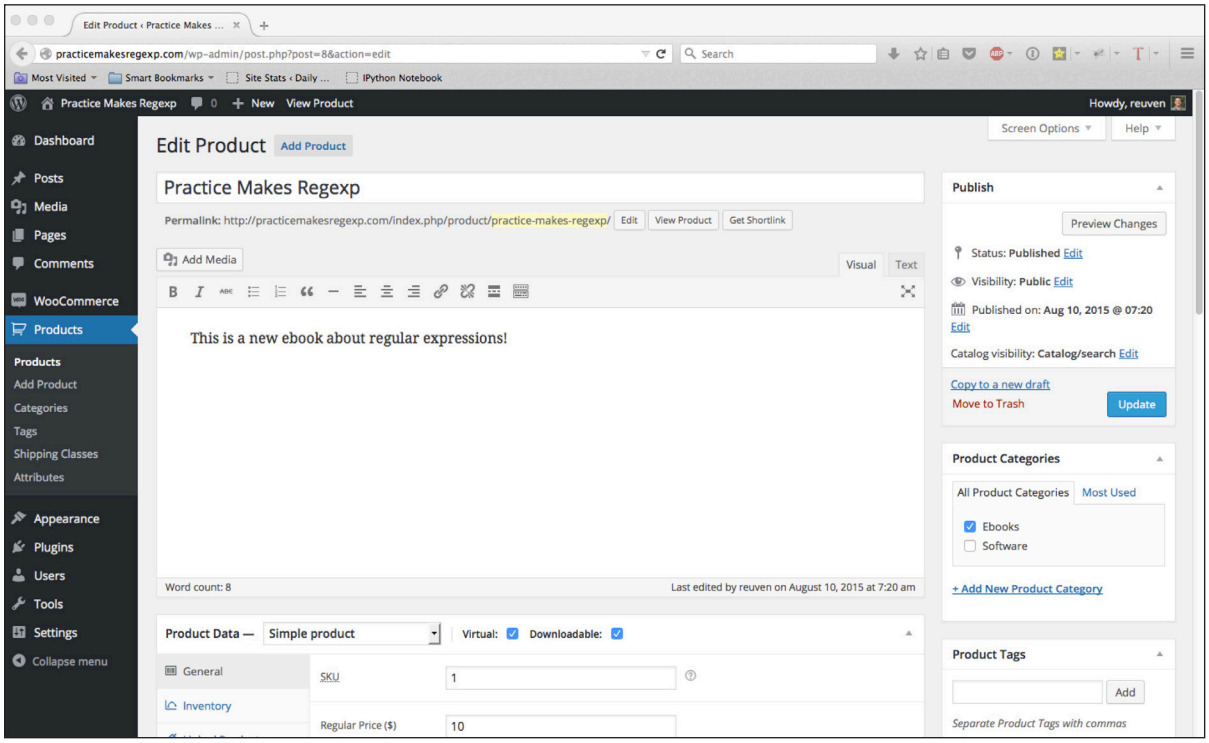

FIGURE 3. Editing a Product

of WordPress categories for your blog post, you can check the boxes for one or more product categories. This means each product can exist within more than one category.

But, this is just the beginning. WooCommerce is packed with features that allow you to create a variety of products and product packages. For example, you can have a "grouped product", which contains a number of other products. Or, you may consider putting together multiple books for a discounted package, or you could create a premium version of a book, containing not only the book itself, but screencasts or other goodies.

For digital products, you'll likely want to click on the "virtual" and "downloadable" check boxes to indicate

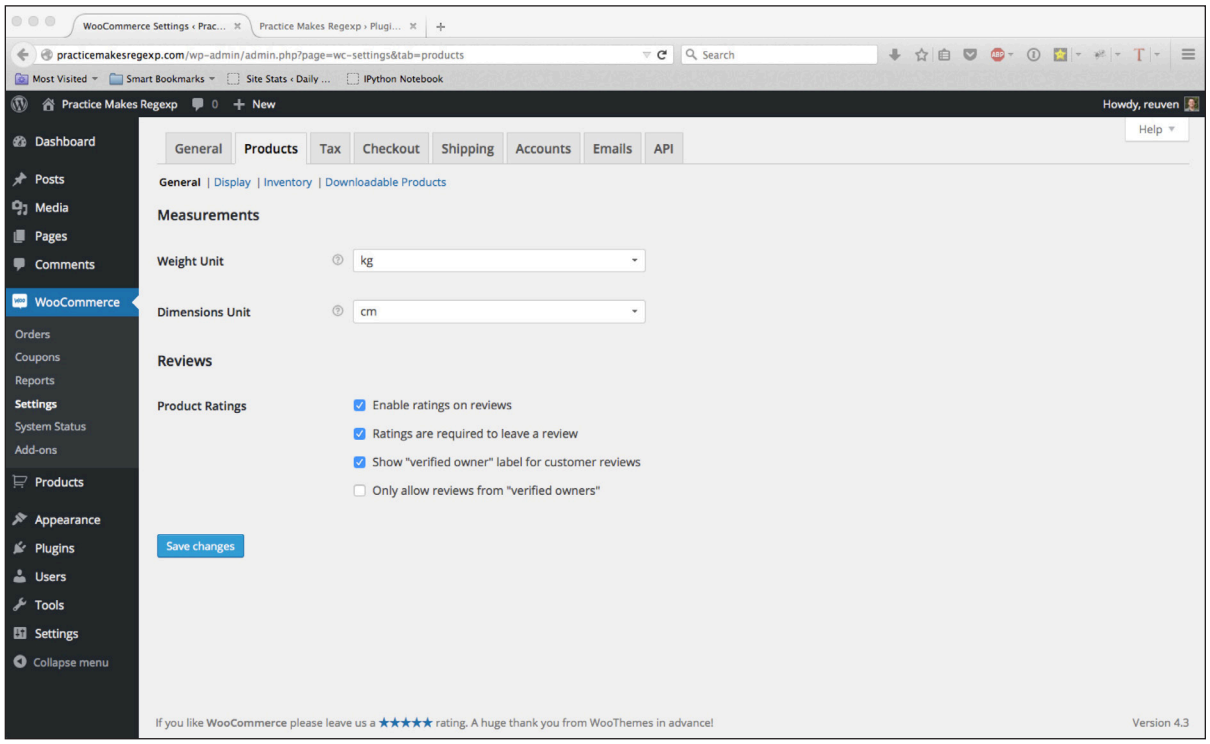

FIGURE 4. **Listing All of Your Products**

that there's nothing physical to ship. Set the SKU (the unique identifier for this product), set a price, and you're done! You even can set a sale price and schedule it automatically—for example, giving users 20% off your book if they buy it during the first week of its release.

Add one or more files that will be part of your digital product, choosing them via URL or from your local filesystem. When you're done with that, click on Publish—the same button you would use to add a blog post is now used to add a new product.

When you're done adding your product, you can go see your shopping cart—but, where? As things are currently defined in WordPress, going to the home page means

seeing the most recent postings. You'll need to configure WordPress to show your shop as the first page. Go to the "Appearance" menu, and from there, choose the "customize" menu option. From that page, choose "static front page", which allows you to choose one of the static pages on the system. From there, you need to indicate that the front page of your site displays a static page, so choose the "shop" static page from the menu to show the e-commerce store. Save the configuration changes (using the "save" button at the top of the page), and go back to your site. You should see your e-commerce shop, with a single product available for download.

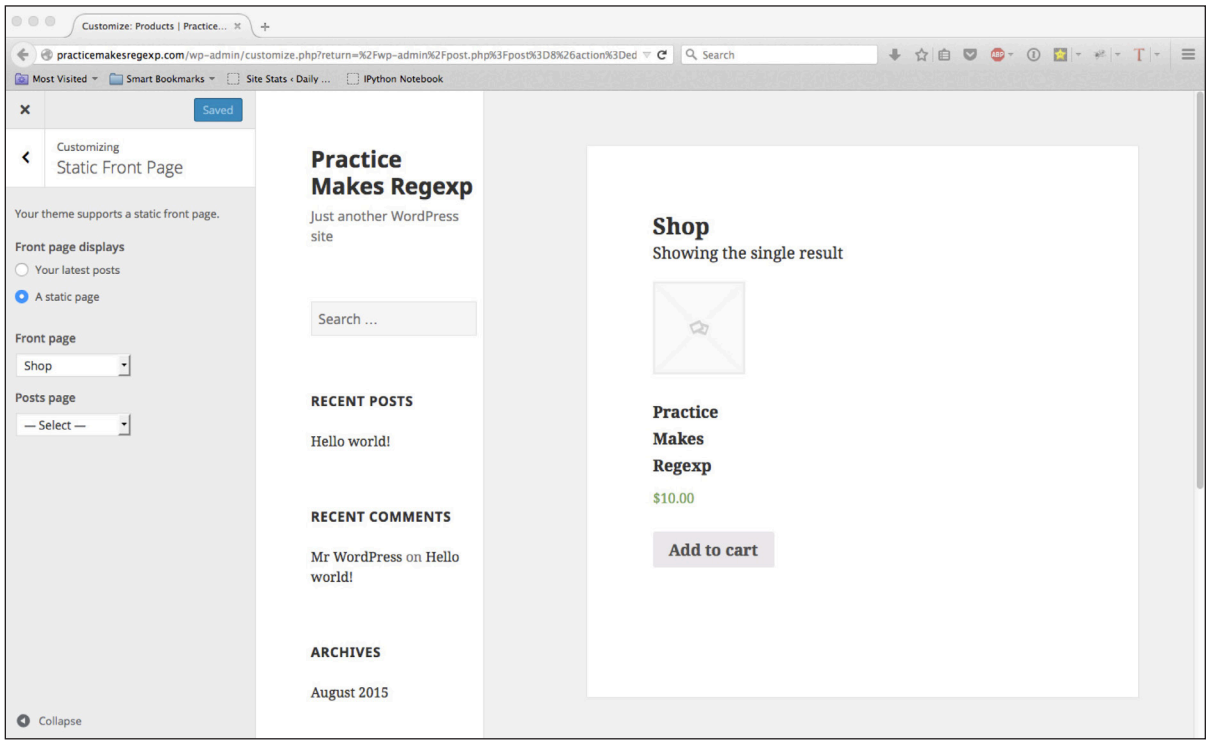

FIGURE 5. **Setting the WooCommerce System to Be the Default Page via WordPress's Dashboard**

Now, it's true that the default WooCommerce site isn't the most beautiful thing in the world. Plenty of themes exist that you can download and install (or hire someone to create), so that your on-line store will look more attractive. But with the above steps in place, you're ready for business. You can add the product to your shopping cart, enter your personal information and pay for your product via PayPal. (You might want to create a coupon offering 100% off of the purchase price, so that you can apply the coupon and not end up spending money on your own product.)

Security

The good news is that your store is up and running.

The bad news is that you need to think about at least two security problems before you start to rake in the money from your on-line store.

The first problem is that you'll almost certainly want to secure your site with a secure certificate (using what's now known as TLS, often referred to as SSL). If you're using PayPal for your payments, then no sensitive information, such as users' credit-card numbers, will be on your site. Your users will be redirected to PayPal to enter that information and will be redirected back to your site when they're done with their purchase. However, even if this is the case, many users might look askance at an on-line shop that doesn't have a secure "lock" icon in the browser.

So no matter what you're doing or what you're selling, you're almost certainly going to want to install a TLS certificate on your server and run WooCommerce

through there. A previous Geek Guide I wrote titled *Apache Web Servers and SSL Encryption* (http://geekguide.linuxjournal.com/content/apache-web-servers-and-ssl-encryption) describes how to install certificates for Apache and provide secure Web services. Those instructions also would be appropriate for the WooCommerce server configured here.

If you're planning to handle payments via a gateway and merchant account on your WooCommerce site, you might well have to get certified as being PCI-compliant (PCI is the "Payment Card Industry" data standard). The credit-card industry has created a set of standards that apply to on-line stores to reduce the risk of fraud and break-ins. If you use PayPal, the onus is on PayPal, and those issues go away. But if you're using a gateway and merchant account, you almost certainly will need to ensure that your system is PCI-compliant.

WooCommerce makes it relatively easy to deal with such requirements; for example, it won't make the classic, horrible, insecure mistake of storing your customers' credit-card numbers in plain text in the database. But I can tell you from experience that making a server PCI-compliant is a bit of a pain, and you should find out from your gateway vendor just how much you'll need to do in order to be compliant with its standards. Much of the compliance has to do with documenting procedures—for example, keeping track of which users have access to your server, which logfiles you're keeping track of (and for how long), having a documented upgrade procedure and regular security tests against your server.

Conclusion

Is that it? The answer, of course, is both yes and no.

From the technical perspective, that's indeed all you need to be up and running. If you follow these instructions, you might well have a running store within an hour or even less.

But having a store, and taking care of the technical side of things, is only the beginning. If you don't market your wares, no one will find them, and your e-commerce dream will vanish overnight. You need to think about how you're going to get the word out about your products, and generate both leads and sales. You'll have to consider how your store looks and what text you display to users. You'll have to think about what sorts of upsells you can do and how you can improve not only the number of visitors who come to your site, but also how many of them buy from you.

In short, setting up the technical side of things is an important step, but it's the first step, not the last one, if you want to sell on-line. You'll need to consider wording, colors and all sorts of other things that programmers usually shy away from, claiming that they're "marketing". Well yes, that is marketing, but marketing is increasingly driven by measurable metrics and statistics, which is something that attracts many programmers.

I've had success selling my ebooks to date, and this has brought me not only some added income, but a great feeling—providing value to people around the world and making their lives better as a result. Selling on-line is an enriching experience in all ways, and now that you see how easy it is to set up a store, what's holding you back?■