Tools and techniques for building a website

NEW PAGES

With readily available Ubuntu tools, you can create and post your personal websites to a server. We'll show you how to get started. **BY JAMES STANGER**

Sophisticated corporate sites include content management tools, server-side scripting, and back-end database systems, but if you just want to start building your own website, the free tools available through the Ubuntu repositories provide all you need to create web pages, podcasts, graphics, and all of the other elements of a solid website.

Even if you're a beginner, Ubuntu's Jaunty Jackalope supports the applica-

HTML Editors

Cream

Cream is a terrific little text editor. Based on vim, Cream is an X Window application that makes it easier for people who want to get right into the code. If you can use any standard Windows application, you can use Cream.

To install it, make sure you have the standard Jaunty Universe repositories enabled and type the following:

```
sudo apt-get install cream
```

Cream only takes up about 2.5MB of disk space on my standard Jaunty system.

Firefox Add-ons

The Firefox web browser can serve as a sophisticated development environment, as long as you install the correct add-ons. Some of my favorites are:

- Firebug [1]: Turns Firefox into an HTML editor.
- Web Developer [2]: Inspects all web page elements, conducts validation, and performs many other functions.
- YSlow [3]: Shows you exactly how long your page will take to download.
- SenSEO [4]: Gives you hints and tips for making sure the pages you create are optimized for the search engines .

If you use Firebug, you get the best of both worlds: a strong text editor and a quasi-WYSIWYG editor. Also, you really don't need to install any other applications.

Bluefish

If you find that using a straight text editor such as Cream is too technical, Bluefish [5] is a nice alternative. Bluefish (Figure 2) is a full-featured HTML editor that provides features such as spell checking, linking to style sheets, and automatically launching various browser types for validation.

Quanta Plus

Quanta Plus (Figure 3) [6] is a KDE-based application that offers image maps, CVS uploads, link validation, and other sophisticated features in addition to basic HTML editing. The user-friendly Quanta Plus organizes HTML code in an intuitive way. For instance, you can select tags from a convenient drop-down menu.

Kompozer

If you just want a WYSIWYG editor that operates more or less like a word processor, the best option might be Kompozer [7]. Formerly known as NVU, Kompozer is easy to use, and it sports some relatively sophisticated features, such as a CSS editor and JavaScript console.

Also, you can put Kompozer into "code mode," wherein the application appears much like a text editor. An intermediate "tag mode" is also available. Tag mode is helpful because it shows you where each of the HTML tags occur on a page. Double-clicking on a tag allows you to edit the code.

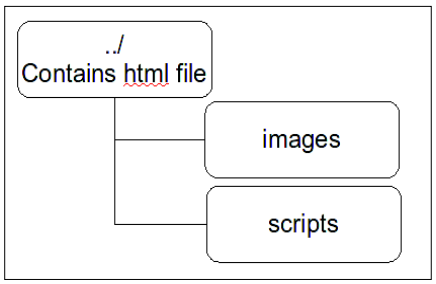

Figure 1: The directory structure of a typical web development environment.

tions you need to be creative and build a personal web presence.

The Big Picture

The task of developing a website generally includes the following steps:

1. Create the HTML from a template.
2. Add images.
3. Test, review, and validate your work.
4. Post your work on a staging server, and then on the production server.

The web page that appears in your browser as a single, unified image might actually consist of several files, including digital images, scripts, and Hypertext Markup Language (HTML) files.

As you probably know, HTML is the markup language at the center of the World Wide Web. An HTML file is a text file with a set of additional codes called *tags* that define how the file will appear in a browser window. An HTML tag can specify a font, a link, an image, or an instruction related to the page layout.

Listing 1: HTML Template

```
01 <!DOCTYPE html
02         PUBLIC "-//W3C//DTD XHTML
   1.0 Transitional//EN"
03         "http://www.w3.org/TR/
   xhtml1/DTD/xhtml1-transitional.
   dtd">
04 <html xmlns="http://www.
   w3.org/1999/xhtml" xml:lang="en">
05 <head>
06 <meta name="Keywords" content="CIW,
   Foundations, Example"/>
07 <meta name="Description"
   content="For the CIW Foundations
   Course"/>
08 <meta http-equiv="Content-Type"
   content="text/html;
   charset=utf-8"/>
09 <title>XHTML Template</title>
10 </head>
11 <body>
12 This is an XHTML file. It validates
   to the XHTML 1.0 Transitional
   standard.
13 </body>
14 </html>
```

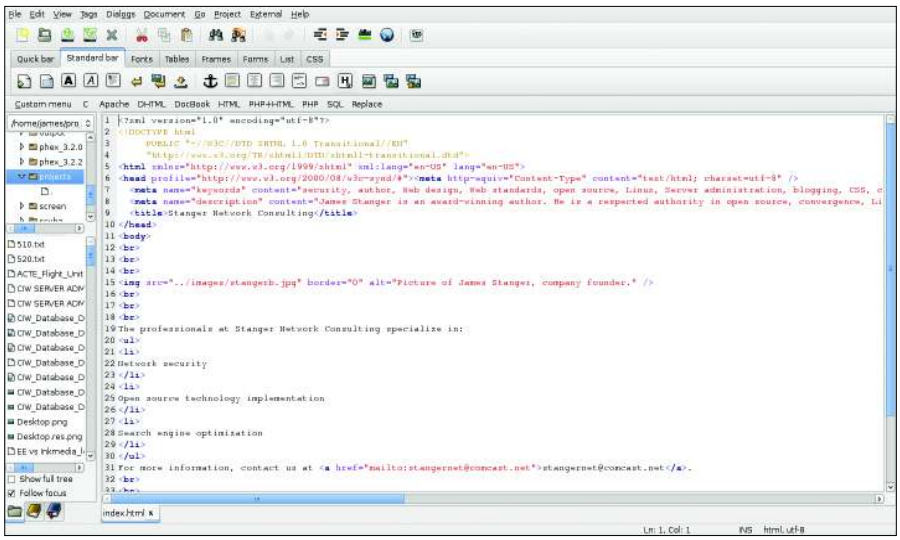

Figure 2: Editing an HTML file in Bluefish.

A full tutorial just on HTML would require a whole book. Several excellent HTML tutorials can be found online. For the purposes of this article, though, keep in mind that you no longer have to hand-code HTML tags into text files. Ubuntu supports several web editing tools that let you work with the page as it will appear to the viewer – without getting hung up in the details of HTML.

The box titled "HTML Editors" describes some editing tools available for the Ubuntu environment. These tools serve the role commonly associated with tools such as Adobe DreamWeaver in Windows and Macintosh environments, except most of Ubuntu's tools are available for free through the Ubuntu repositories. For more on finding and installing software in Ubuntu, see the article on package management in the Discovery guide.

Establishing a Development Environment

As you begin building your website, it is a good idea to create separate directories for your HTML pages, images, and scripts (Figure 1). Organizing your files into a discrete, well-defined directory structure helps ensure that your pages will render properly in a web browser after they've been transferred to a web server system.

Rather than working on a live Internet site, most developers do their web development and testing on a staging server that is as close as possible to the final web server environment. Working on the staging server lets you review and debut code in a nearly real environment, but if you make a mistake, the general public

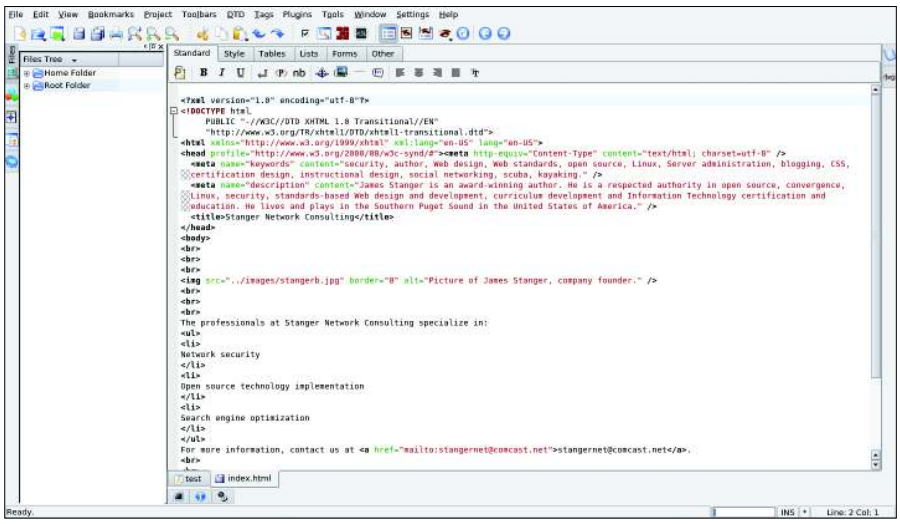

Figure 3: Editing an XHTML page in Quanta Plus.

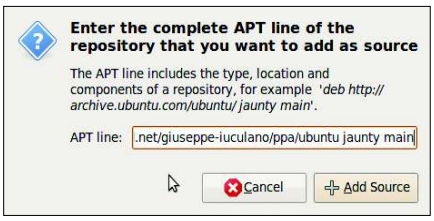

Figure 4: Entering a repository in Synaptic. If the APT line is too long for the box, just keep typing.

won't see it. Once you are satisfied with the web pages, you can upload the directories to a production server.

Templates and HTML

Whenever I create a web page, I always start with a standard template file. The template file contains standard information that allows me to create a consistent set of pages. Listing 1 shows a barebones HTML template. As you can see, the template provides a standard form for calling out parameters such as the HTML version, the language, and the character set. Other statements let you designate keywords and descriptive text that will be associated with the site.

The very simple template shown in Listing 1 provides only a fraction of the possible options, but it is a good beginning. If you are looking for a more sophisticated alternative, try the Open Source Web Design [8] or Open Source Templates [9] websites.

The *!DOCTYPE* statement, which specifies the HTML standard, is an important element of the template. Often I'm asked which version of HTML or XHTML people should use. Although I prefer XHTML 1.0 Transitional – it is supported by the majority of web browsers, is perfect for data-oriented social networking applications, and is relatively current – to be honest, you can choose any version of HTML you want, including good old HTML 4.01. Just pick a standard and use it consistently.

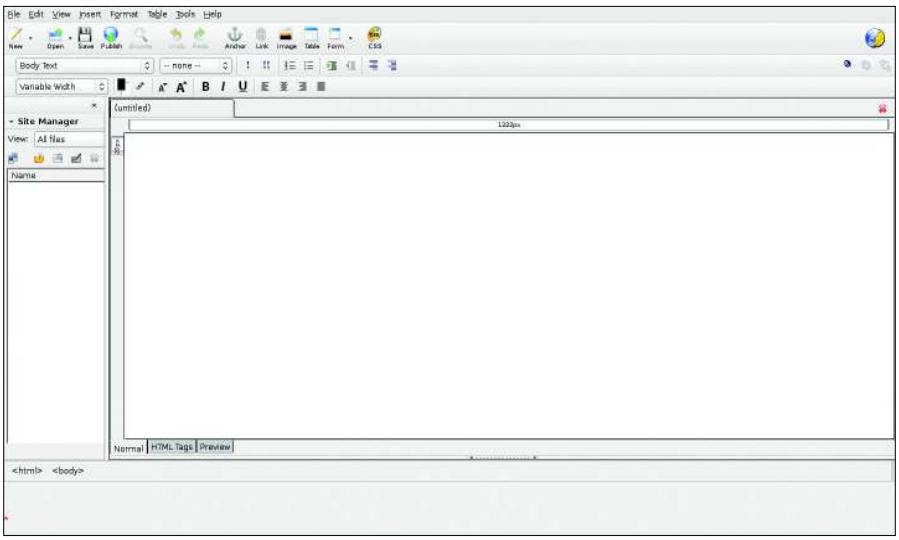

Figure 5: Kompozer opens in WYSIWYG mode.

Because an HTML file is essentially a text file, you can create and edit HTML files with an ordinary text editor, such as vim, emacs, or the gedit editor available through Ubuntu's user interface (*Accessories* | *Text Editor*). A pure text editor, however, doesn't check your code or provide any particular coding advice.

An HTML editor is specifically designed for the task of creating web pages. HTML editors inform you about the quality of the code you're entering. An HTML editor will also make it easy for you to insert standard HTML tags.

Several HTML editors are available for Ubuntu, and the ideal choice is often a matter of personal taste (refer to the box labeled "Choosing an Editor").

Kompozer

For this article, I'll focus on setting up a page using Kompozer. The other editors are similar – see the documentation for your own editing tool. Kompozer doesn't appear in the default Ubuntu user interface, so you'll have to add it. If you're comfortable working from the command line, you can use the apt-get utility. If not, you can add Kompozer through the Synaptic package manager.

To open Synaptic, choose *System* | *Administration* | *Synaptic Package Manager* in the Ubuntu main window. To access Synaptic, you'll need to enter the administrative password (the password you defined when you installed the system).

In the Synaptic main window, choose *Settings* | *Repositories* and select the tab labeled *Third-Party Software*. In the *Software Sources* dialog box, click the *Add* button. Synaptic will prompt you to enter the APT line. Enter the following (as shown in Figure 4):

```
deb http://ppa.launchpad.net/ &
giuseppe-iuculano/ppa/ &
ubuntu jaunty main
```

Next, click the *Add Source* button to add the repository, then close the *Software Sources* dialog box and click *Reload* in the Synaptic main window. Synaptic will

Choosing an Editor

A few important qualities to look for in an editing tool include:

- Standards compliance: An HTML editor should support open standards and make it easy to choose a standard, from HTML 4.01 to XHTML 1.0 Strict and beyond.
- Cascading Style Sheets (CSS) support: CSS is a method for adding style elements to web pages. As you advance to more sophisticated web development assignments, your job will be much easier if you're using an editor that supports CSS.
- File publishing: A good HTML editor should provide a means for uploading web pages to the server with a minimum of fuss.
- Ease of use: An application that gets you up and running quickly might be worth the sacrifice of multiple feature sets.

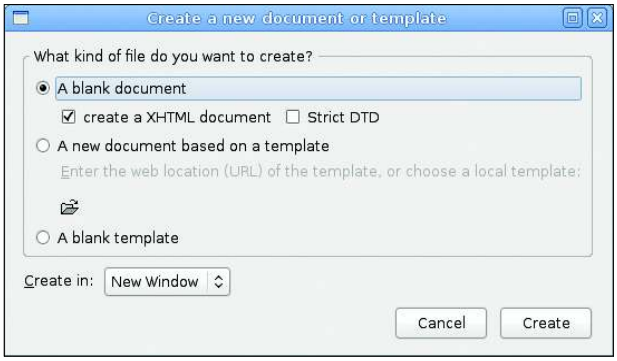

Figure 6: Creating a new document in Kompozer.

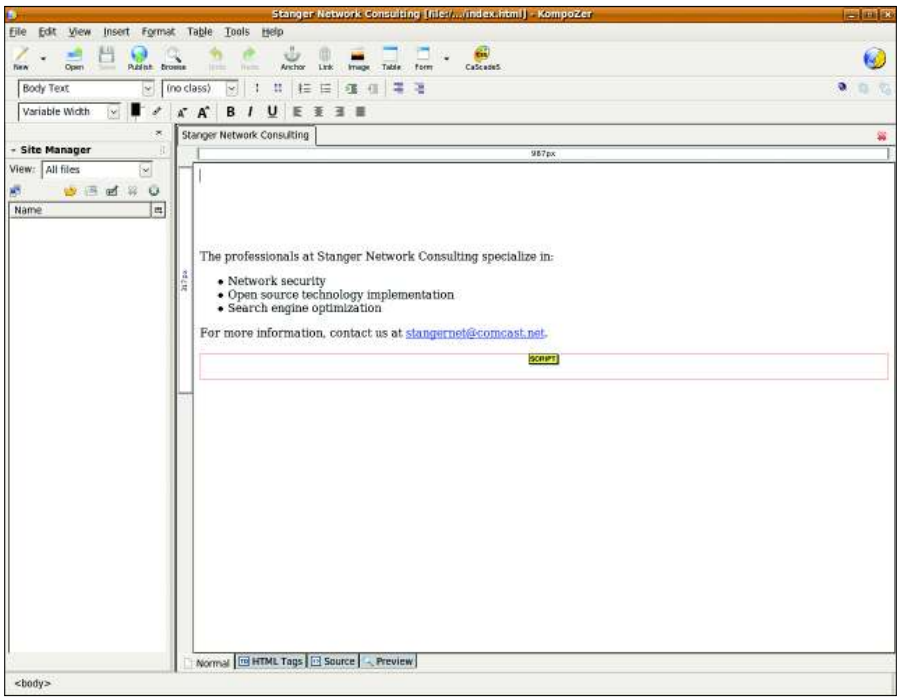

Figure 7: Creating a bulleted list in Kompozer.

add the contents of the repository to the package list.

If the reload is successful, enter *kompozer* in the *Quick Search* box. When *kompozer* appears in the package list, select the checkbox to add the package and click the *Apply* button. Synaptic will add Kompozer to your Ubuntu system. (For more information on adding new applications to your Ubuntu system, see the article on adding packages in the Discovery guide.)

Lauching

On standard Ubuntu systems, Kompozer will not appear automatically in the menu once you add it. To start Kompozer, launch a terminal window by selecting *Applications* | *Accessories* | *Terminal* in the Ubuntu main window and type *kompozer* (all lowercase – remember that Ubuntu is case sensitive) at the command prompt. Kompozer starts with the user-friendly interface shown in Figure 5. *Note:* Some early versions of Jaunty have experienced issues with Kompozer. These issues will likely be cleared up by the time you read this, but the Ubuntu forum has posted some information for early adopters [10].

Spell Checking

Right now, Kompozer doesn't have a good spell checking feature. If you're stumped for decent spell checking in Kompozer, consider copying and pasting text from Kompozer into OpenOffice.

Creating a Document and Formatting Text

To begin a new document in Kompozer, choose *File* | *New*. Now you'll be presented with the *Create a new document or template* box. To create an XHTML 1.0 Transitional document, edit the dialog box so that it looks similar to the box shown in Figure 6.

Once you click *Create*, you'll then be presented with a blank document. Just start typing inside the document as you would any word processor. As you type in some text, you can also apply formatting, including bullet points, headers, and so forth. To add bullets, create three new lines of text, then highlight them and go to *Format* | *List* | *Bulleted*.

The list you've just created is now bulleted, as shown in Figure 7.

As you explore the Kompozer menus, you will find other options for aligning, resizing, and italicizing text.

Adding an Image

Most web pages include graphic images as well as plain text. With the use of an image processing tool, you can add a photo image or create a graphic element for the site. GIMP is a popular graphics tool that is accessible through the Ubuntu Applications menu (*Applications* | *Graphics* | *GIMP Image Editor*). For a quick look at how to build a simple graphic header for your website, see the sidebar titled "Creating a Simple Header Bar."

To add an image in Kompozer, place your cursor in the document where you want the image to reside. For example, place the cursor near the top of the page. Then, choose *Insert* | *Image*, and browse to an image you have previously created and saved.

Now enter the location of the image and some alternative text that describes it. The alternative text helps search engine bots organize the information on the site, which could increase the search rank for the page. Also, screen reading tools use the alternative text to describe the contents of the image to visually impaired users. Once you have inserted the image, it will appear in the Kompozer

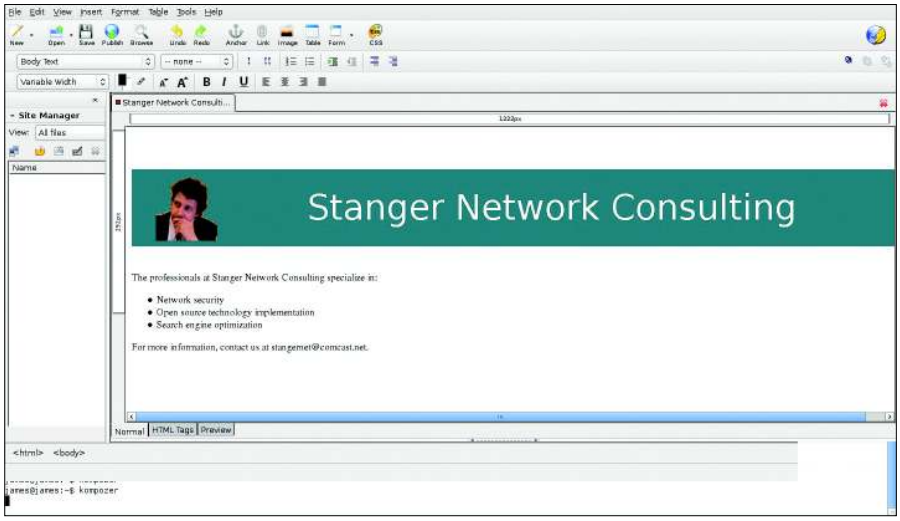

Figure 8: A web page in Kompozer after inserting an image.

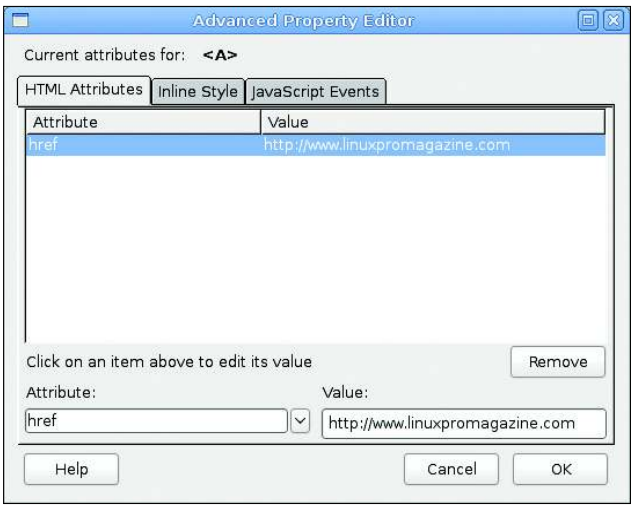

Figure 9: Inserting a link in Kompozer.

window, as shown in Figure 8. If the image doesn't appear, try viewing the code in a browser by clicking on the *Browse* button.

Hyperlinks

Hyperlinks are the live links familiar to all web users. If you click on a link, the browser jumps to another web page or marker. To insert a hyperlink in Kompozer, first highlight the text you want to turn into a link, then either click on the *Link* button or go to *Insert* | *Link* to bring up the *Link Properties* window. Next, click on the *Advanced Edit* button to bring up the Advanced Property Editor window, shown in Figure 9.

In the *Attribute* field in the lower-left portion of the window, use the drop-down box to select *href* (the attribute associated with a hyperlink). Then, in the *Value* field, enter the URL you want to link to.

To return to the page to see that the text you've highlighted is now a hyperlink, click the *OK* button twice.

HTML Validation

After you finish creating your web page, you're still not ready to post it. First you need to prove that the code you generated is compliant with the World Wide Web Consortium (W3C) standards. In addition to the troubleshooting benefits of validation, validated pages render better in more browsers, and search engines tend to give validated pages a higher rank.

Various applications and services are available to help you get your code to pass W3C muster, including The W3C Markup Validation Service [11], Weblint [12], and Firefox add-ons such as Relaxed [13] and Total Validator [14].

To validate your web page with the W3C Markup Validation page, point your web browser to *http://validator.w3.org*.

The Validation page gives you the choice of entering a URL to a website or uploading a file. Because you haven't published this page yet, I'll assume that you're going to upload a file. As shown in Figure 10, click on the *Validate by File Upload* tab, then click on the *Browse* button.

To check a file, highlight it, click *Open* to return to the W3C page, then select *Check*. If your page was created with proper HTML or XHTML code, you will see a screen similar to that shown in Figure 11.

If your code has any compliance issues (and often it does), you will see a list of errors. Troubleshooting these errors takes quite a bit of time because the W3C service is not particularly intuitive with its hints and directions. Nevertheless, validation is an important step in ensuring that your web page will load properly in a variety of browsers.

But don't be happy just because your pages validate: It is also important to check the pages directly by opening them in a variety of web browsers.

Getting to the Server

Once you have created and tested your web page, it is time to post it on the staging server. In most cases, you won't be building your own whole new server but instead will be uploading files to an existing server system. Many Internet providers offer a small amount of web server space for subscribers, and most companies have access to servers that support a permanent web presence.

If you want to use the Apache web server included with Ubuntu to test and experiment with your HTML files, issue the following command to start Apache:

```
sudo /etc/init.d/apache2 start
```

The */var/www/* directory is the default location for all files on the web server. All you have to do is place any file you want the server to render in this direc-

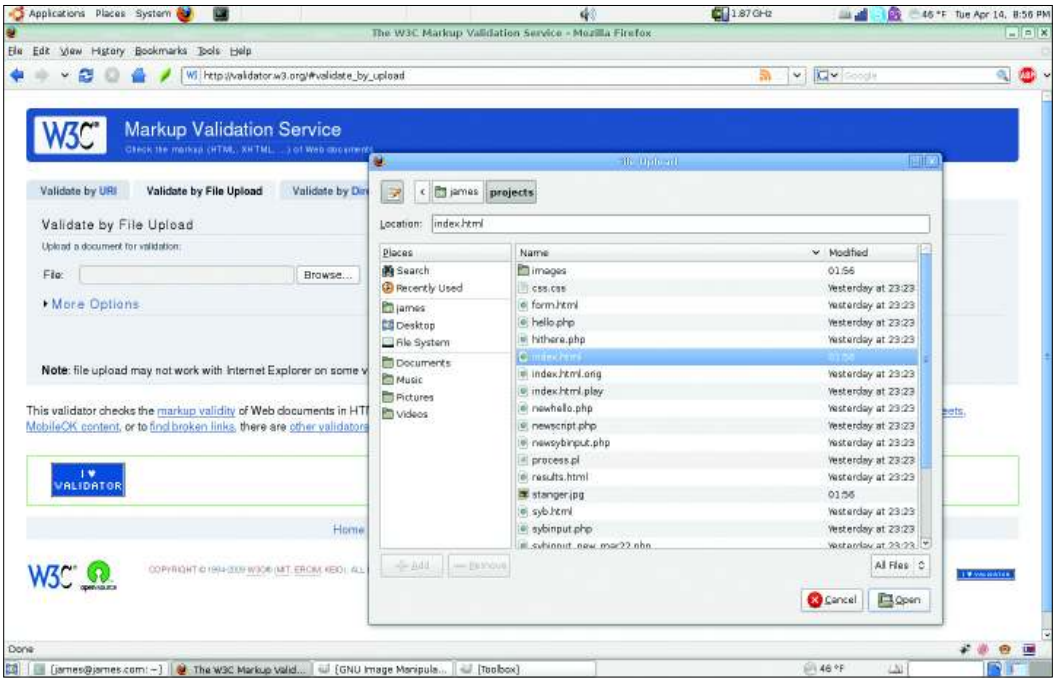

Figure 10: Choosing a file for validation.

tory. To place files in this directory, you need root privileges.

The default document is *index.html*. The server will automatically present this file to anyone who points their web browser at the server instance, so web developers typically give this name to the file that will serve as the homepage.

One popular way to copy the web files to the server is by entering the *cp* command in a terminal window:

```
sudo cp -r ➥
projects/* /var/www
```

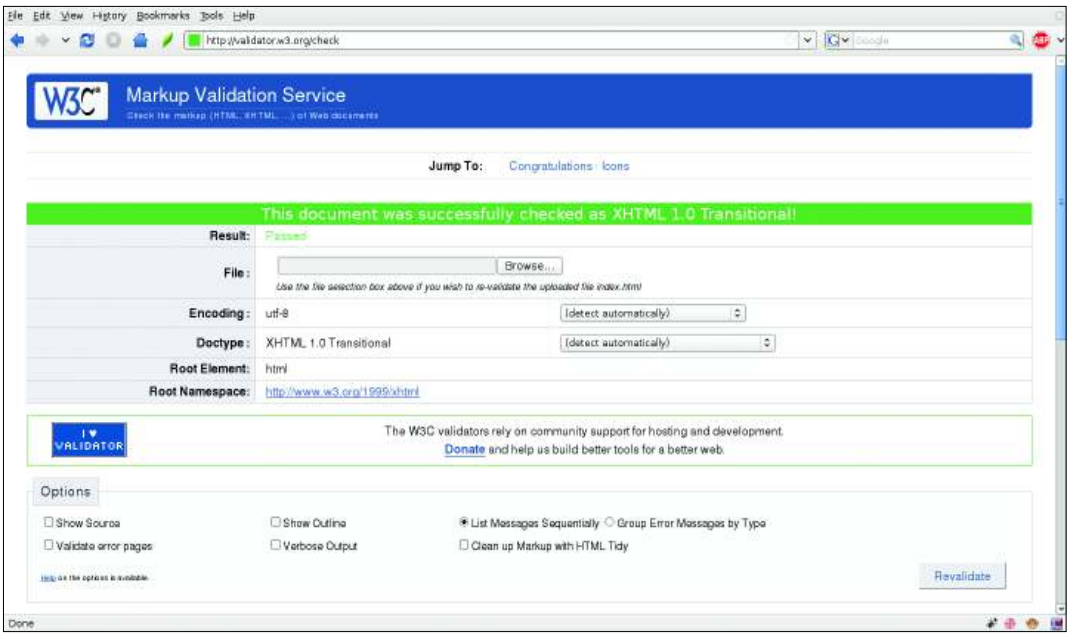

Figure 11: HTML code that has passed validation.

Because the root account owns the */var/www/* directory, you need to use the *sudo* command.

If you need to send the files across the Internet, you can use ftp or even email to deliver the pages to the server.

Most HTML editors have a feature that lets you publish your site directly to the web server from within the editor. This feature is basically nothing more than an ftp client that you can configure to publish pages with one click. In Kompozer, for example, all you have to do is hit the *Publish* button on the taskbar or go to *File | Publish*. When you enter the necessary information in the Publish Page window, you are on your way to publishing the site.

Conclusion

Ubuntu provides a powerful, free development platform for building websites. In today's economy, the tools described in this article are more than enough for even a seasoned developer to create compelling, industry-standard web pages.

The next step for you to take is to finish establishing your development environment. Experiment with the tools that I've introduced here. Then, take a look at the myriad sites and education offerings out there about web design. Sites such as *ciwcertified.com*, *how-to-build-websites.com*, and others can provide you with excellent ways to further your skills. ■

Creating a Simple Header Bar

GIMP is the de facto open source standard for creating standard RGB (Red Green Blue) files. GIMP is not exactly Adobe Photoshop, but it's just about good enough.

GIMP supports several image file formats, including JPEG, GIF, and PNG.

The steps I used for creating the simple header bar shown in Figure 8 are as follows:

1. In the GIMP File menu, choose *New* and then specify the pixel dimensions. The size for this header is 1257x124 pixels.
2. Click on *Advanced Options* and then specify the *X* and *Y* resolution. Here, I will use 72 by 72 dots per inch (dpi) for the header, which is not ideal for images containing faces but is suitable for basic images used for navigation or background. If you specify a higher resolution, the image might be clearer, but the page will take more time to download and render. Even in these days of increased bandwidth, a slight increase in download time could still make users impatient.
3. To specify a background color, choose the "bucket fill" tool and a color.
4. For the header, select the text feature and a color. Here, I will use white.
5. A second, already-existing image can be added to the image you are creating. For this example, I will drag a transparent GIF photo image onto the bar. "Transparent" means that the background of this new image will be the same color as the image behind it.
6. Now save the image as a JPEG by simply adding the *.jpg* ending to the file name.

INFO

[1] Firebug: *https://addons.mozilla.org/en-US/firefox/addon/1843*

[2] Web Developer: *https://addons.mozilla.org/en-US/firefox/addon/60*

[3] YSlow: *https://addons.mozilla.org/en-US/firefox/addon/5369*

[4] SenSEO: *https://addons.mozilla.org/en-US/firefox/addon/9403*

[5] Bluefish: *http://bluefish.openoffice.nl*

[6] Quanta Plus: *http://quanta.kdewebdev.org/*

[7] Kompozer: *http://www.kompozer.net*

[8] Open Source Web Design: *http://www.oswd.org*

[9] Open Source Templates: *http://www.opensourcetemplates.org*

[10] Kompozer issues: *http://ubuntuforums.org/showthread.php?t=1094500*

[11] The W3C Markup Validation Service: *http://validator.w3.org*

[12] Weblint: *http://www.w3.org/Tools/weblint.html*

[13] Relaxed: *https://addons.mozilla.org/en-US/firefox/addon/3939*

[14] Total Validator: *https://addons.mozilla.org/en-US/firefox/search?q=html+validator&cat=all*