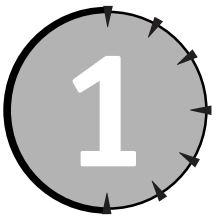
SESSION



Installing Apache

Session Checklist

- ✔ Understanding why you should use Apache
- ✔ Gathering required materials
- ✔ Installing Apache
- ✔ Testing the installation

30 Min. To Go

This book shows you how to knit together Web server, database server, and scripting technologies. These three technologies enable you to deliver powerful and dynamic content via the Web. Before you can begin using the technologies, you need to install all three components, starting with the Apache Web server. These first three sessions walk you though the process of installing the technologies, testing each, and testing their interactions to ensure that you are ready to start working with each.

Why Use Apache?

Apache powers the Web. Although this seems a grandiose claim, there is a lot of truth to it. Recent surveys show that an overwhelming number of Web sites run Apache as their Web server. That being the case, why do all these Web sites use Apache?

- Apache is free.
- Apache is open source.
- Apache is cross-platform.
- Apache is continually undergoing rapid development.
- Apache is powerful, yet modular.

Apache Is Free

Apache is a full-featured, powerful Web server available absolutely free. Because the Apache Software Foundation is not deriving revenue from the Apache server, however, it cannot afford to offer robust technical support. Amenities such as phone or online support are not included with Apache. Abundant documentation is available, but support at the level you may be used to with commercial software is not.

Apache Is Open Source

You can get the source code for Apache and modify it to your heart's content. Most people don't use the source code to modify how Apache works; they use it to modify how Apache is built — that is, what options are compiled into the server. If you need a mean, lean server, you can recompile the source code to create a custom server with only the options you need. That said, if you ever find a problem or need to make a rudimentary change to the Apache source code, you can.

The concept of *open source* software is not new, but the idea can be rather intricate. For more information on Open Source software, visit GNU.org and read the various licenses: `www.gnu.org/licenses/licenses.html`**.**

Apache Is Cross-Platform

Apache is available for multiple platforms, including the following:

- Unix
- Linux
- Windows (9*x* through XP, although server versions — NT/2000 — are preferred)
- Novell NetWare
- Mac OS X (BSD under the GUI)

Besides a few minute details, such as the placement of its files in the file system, Apache operates the same on all of the aforementioned platforms.

Apache Is Continually Undergoing Rapid Development

Apache is maintained by the Apache Software Foundation and is under continual development and improvement. Bug and security fixes take only days to find and correct, making Apache the most stable and secure Web server available.

The relative stability and security of any Web server depends on the system administrator as much as, if not more than, the underlying software.

Another advantage of rapid development and releases is the robust feature set. New Internet technologies can be deployed in Apache much more quickly than in other Web servers.

Apache Capabilities

Apache gets its name from the way it was originally developed. Originally, the server was made of several components or "patches," making it "a patchy server."

Apache continues to implement its features with distinct pieces, or *modules*. Utilizing a modular approach to feature implementation enables Apache to be deployed with only the amount of overhead necessary for the features you want. It also facilitates third parties developing their own modules to support their own technologies.

Apache supports almost all Internet Web technologies, including proprietary solutions such as Microsoft's FrontPage Extensions. Apache supports all manner of HTTP protocols, scripting, authentication, and platform integration.

Visit the Apache module Web site (`http://modules.apache.org`) for information on the modules included with Apache and the registered third-party modules.

For our purposes, we care about the following capabilities:

- Robust HTTP delivery
- Configurable, reliable security
- Integration with PHP
- CGI and other scripting integration

Gathering Required Materials

Everything that you need to install Apache can be found at the Apache Web site, at `www.apache.org`. You can download the source and/or binary files for Unix/Linux installations or a binary install package for Windows. The main Apache.org Web site is shown in Figure 1-1.

Apache is also packaged for most Linux distributions. For example, Red Hat maintains an Apache Red Hat Package Manager (RPM), which can be used to install Apache on a Red Hat system. If you don't need the absolute latest version of Apache and don't need it configured a particular way, it is worth visiting the Web site for your distribution to download an appropriate Apache package.

The Apache Software Foundation maintains several different projects — the HTTP Server Project being the most prominent. The main page for the HTTP server can be found at `http://httpd.apache.org`. Various links from this page lead to source and binary code downloads, documentation, and other resources. The main HTTP Server Project page is shown in Figure 1-2.

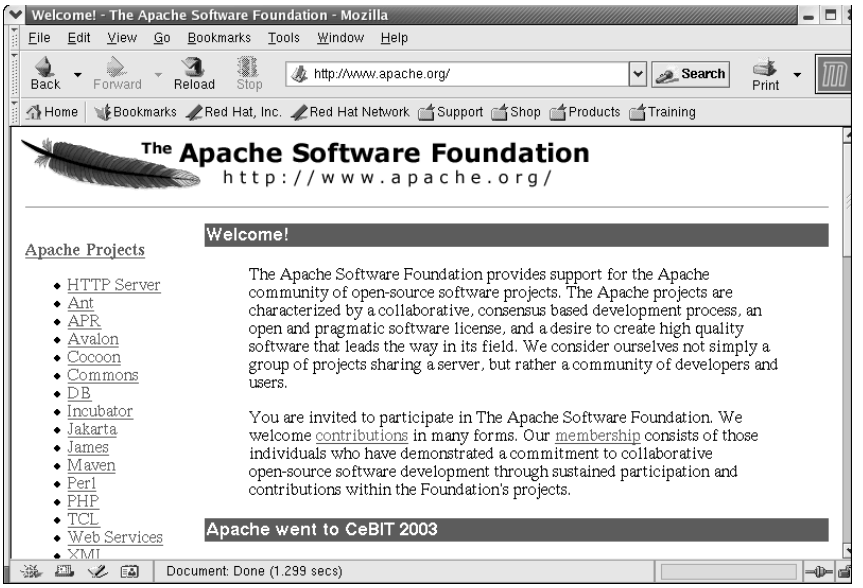


Figure 1-1 *The Apache.org Web site*

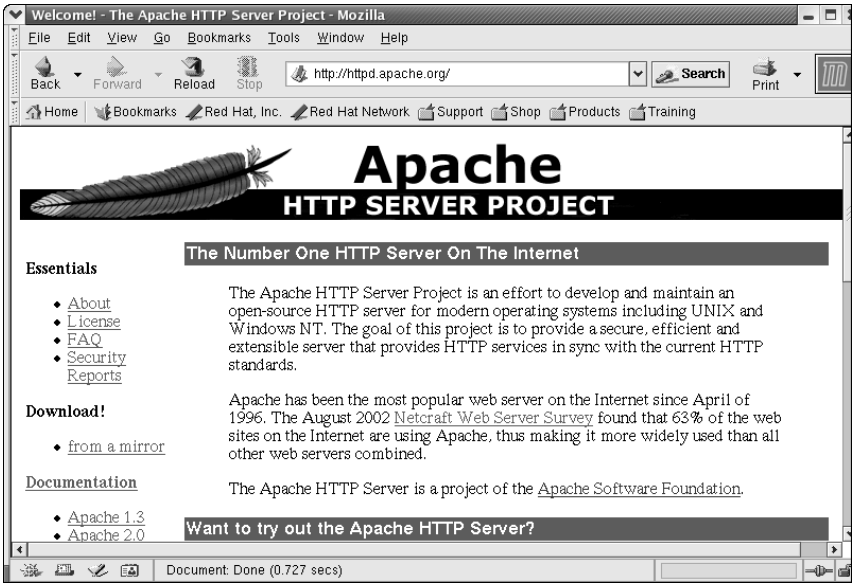


Figure 1-2 *The Apache HTTP Server Web site*

At the time of this writing, the current Apache server version is 2.0.47.

Documentation on Apache is also available from the Apache Web site. Downloading the PDF version of the documentation for later use is worth the time.

At the time of this writing, the PHP project is still issuing warnings about using PHP 4.*x* on Apache 2.0 in a production environment. Although I've not seen any issues combining the two, a warning from the project should not be taken lightly. If you are going to use PHP and Apache in a production environment, you might want to consider running Apache 1.3.*x* instead of Apache 2.0.*x*.

Windows Downloads

Windows users should follow the download links to the Windows binary install (currently available as an MSI Installer — a special program for installing packages on Windows). Download this file to a temporary folder on your local hard drive.

If you need a specially compiled version of Apache, the source files for the Windows version are also available. You need to have a suitable C++ compiler installed — Microsoft C++ version 5.0 or later is recommended. Other tools and special configuration options are also necessary to compile Apache. You can find additional information on compiling Apache for Windows at `http://httpd.apache.org/docs-2.0/platform/win_compiling.html`**. This session covers installation of the Windows binaries only.**

Linux Downloads

Many methods for installing Apache on Linux are available; the method that you choose depends on your answers to the following questions:

- What distribution of Linux are you running?
- Does your distribution have a package with a recent version of Apache?
- Do you need to compile your own version for compatibility or capability reasons?

If you are running one of the major Linux distributions (Red Hat, Mandrake, United Linux, Debian, and so on), chances are good that a recent copy of Apache is packaged for your distribution. A recent installation of Red Hat 8.0, for example, contains Apache version 2.0.40 release 8 on CD #2 in RPM form (`httpd-2.0.40-8.i386.rpm`). After installation, you can use the Red Hat Update Agent to update Apache to 2.0.40 release 11, the latest version of Apache packaged for Red Hat.

If your Linux version does not have a recent version of Apache packaged, you can download the generic binary version for installation on your system. An `Other files` link is available from the Download page on the Apache HTTP Server Web site. Follow the resulting links (starting with `_binaries_`) to identify and download the version that most closely matches your distribution and configuration. If you are using Red Hat version 7.3 on a Pentium III or IV processor, for example, you want to download the following file:

```
httpd-2.0.40-i686-pc-linux-gnu-rh73.tar.gz
```

Download and read the `README` **file associated with the archive before you download and install the archive. The** `README` **file contains the options the binary was compiled with. (Notice that the term *archive*, as used here and throughout this book, evolves from the Unix world, where archives of collections of files were written to magnetic tape. The utility** `tar` **stands for *Tape ARrchive*. Today, it simply means a collection of files bundled together for a particular purpose.)**

If you need a specially compiled version of Apache, you can download the source and compile Apache yourself.

20 Min. To Go

Installing Apache

Despite the fact that Apache is a full-featured HTTP server, installing it is actually as simple as installing a regular application. Because it is a server, you face many implications on system security after Apache is installed. Security issues are covered in depth in Session 5.

Installing Apache on Windows

If you are installing Apache on Linux, you can skip this section and move on to the section "Installing Apache on Linux from Packages," later in this session.

To install Apache on Windows, you need to download the Windows binary installer, as covered in the section "Windows Downloads," earlier in this session. This file is usually named as follows:

```
apache_2.0.*-win32-x86-no_ssl.msi
```

The asterisk (*) indicates the minor version number. At the time of this writing, the minor version is 45.

Although you *can* run Apache on Windows 9*x*/Me, doing so is not advised. Windows NT/2000/XP provides a substantially more stable and secure base for any server application. The following instructions were performed on a copy of Windows 2000 Professional.

Place this file in a temporary directory and follow these steps to install Apache:

1. Log into Windows as an Administrator.
2. Use Windows Explorer to locate the file on your hard drive, as shown in Figure 1-3.
3. Double-click the installer to begin the installation.
4. The Apache installer performs like many other Windows installers, using a Wizard-like approach, as shown in Figure 1-4.

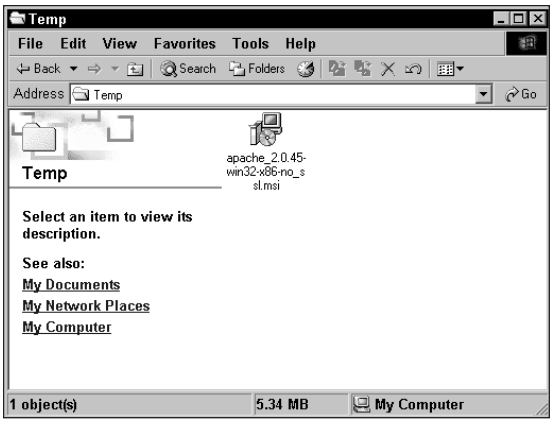


Figure 1-3 *Find the installer on your hard drive.*

Figure 1-4 *The Apache installer runs like many Windows installers, presenting information and settings in Wizard-like form.*

5. Click the Next button, read and confirm your acceptance of the license agreement, and click Next again.
6. The next Wizard window displays useful information about running Apache on Windows for anyone new to the process. Read it before clicking Next.
7. The next window enables you to specify server information. This information should be populated from information already present in Windows, as shown in Figure 1-5. The information should be okay as is; review it before clicking Next. If necessary, modify the information to suit your needs.

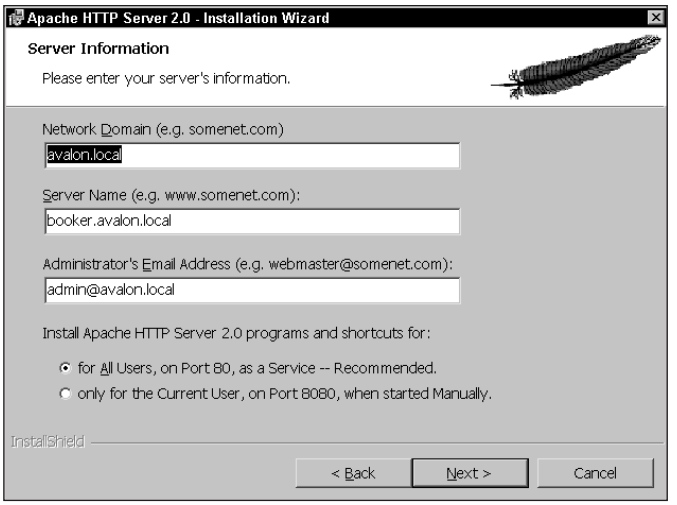


Figure 1-5 *The Server Information window enables you to specify the domain, server name, and how the server is to run.*

If you need to edit the dialog box, set the Network Domain field to the name of your domain. In this case, you are running Apache on an internal network, so you specify `.local` as your top-level domain instead of a `.com`, `.org`, or other top-level domain. The Server Name field should be the machine name (or `www`), complete with the fully qualified domain information. The last option enables you to control how the server is run. You're best off accepting the default, `for All Users, on Port 80, as a Service—Recommended`. Click Next after the settings are complete.

This information can be changed later by editing the Apache configuration files.

8. The next window enables you to select to install Apache in the typical location with the typical components, or specify a custom installation. The typical installation installs Apache in the directory `C:\Program Files\Apache Group`. (The drive letter may vary, depending on your individual installation.) The typical installation installs the Apache binary files and documentation but not the headers and libraries. You should select the typical installation unless you need to change any of these settings. Click Next to continue the installation.
9. You are given a chance to change the default directory where Apache is installed. Accept the default by clicking Next.
10. A confirmation window gives you one more chance to correct any installation options. Click Back to change any options or Install to begin the installation.
11. After the installation completes, a completion window is displayed. Click Finish to end the installation program.

After Apache is installed, the server starts automatically. You can verify that the server is running by checking the Apache Service Monitor icon in the system tray, as shown in Figure 1-6.

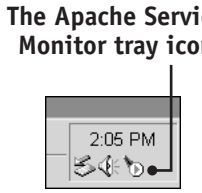


Figure 1-6 *The Apache Service Monitor tray icon displays the server's status.*

If the icon has a green arrow, the server is running. If the server is not running, a red dot replaces the arrow. You can double-click the icon to display the Apache Service Monitor. Using this monitor is covered in Session 4.

Unless you also need to install Apache for Linux, you can skip the next section and resume at the section "Testing the Installation," later in this session.

Building and Installing Apache for Linux (from Source)

If you need to compile Apache, you need an appropriate source archive. You can download one of the standard archives from Apache.org or obtain one from your distribution vendor. This session covers compiling from the source files on Apache.org. If you have a binary archive or package to install, you can skip this section.

The source files are available from the Apache HTTP server download page as a gzipped tarball or a compressed tarball. Download the appropriate file into a temporary directory or the typical location for source files on your system (usually `/usr/src`).

***Tarball* is a common name for an archive packaged with the `tar` utility and is a common means of distributing software, much like ZIP files are used for Windows. In their raw form, tarballs do not incorporate compression; they must be compressed by using special tools such as `gzip` or by using the compress options in the `tar` utility.**

You can use several methods to unpack the source files; the methods available depend on the utilities you have installed and the file you downloaded.

If you downloaded the gzipped tarball file (with a `tar.gz` extension), you should use the `gzip` utilities to unzip the file before it is unpacked with tar. The following command line, performed in the directory where the source archive is located, uncompresses and unpacks the gzipped archive:

```
tar -zxvf httpd-2.0.*.tar.gz
```

If you downloaded the compressed tar file (with a `tar.Z` extension), you can uncompress and unpack the archive with the following command line:

```
tar -zxvf httpd-2.0.*.tar.Z
```

Either of the preceding methods results in unpacking the Apache source code into an appropriately named directory — namely, `httpd-2.0.*`, where * is the minor revision number.

Uncompress and unpack the archive in the directory reserved for source files (for example, `/usr/src`). This results in a `httpd-2.0.*` directory under the source directory (for example, `/usr/src/ httpd-2.0.*`) in which the Apache source files are stored.

The instructions for building and installing Apache are contained in the file named `INSTALL` within the source file directory. Essentially, the build and install consists of three commands, as follows (as taken from the `INSTALL` file):

```
./configure --prefix=<directory>

make

make install
```

These commands should be issued in the directory where the source files were expanded (`httpd-2.0.*`), and the commands should be run as root.

The first line configures Apache for compilation on your system. The `<directory>` should be replaced with the directory in which you want to install Apache. If you wanted to install Apache into the directory `/usr/local/apache2`, for example, you would use the following command line:

```
./configure -prefix=/usr/local/apache2
```

The output from the `configure` command resembles the following:

```
checking for chosen layout... Apache
checking for working mkdir -p... yes
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu

Configuring Apache Portable Runtime library ...

checking for APR... reconfig
configuring package in srclib/apr now
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
Configuring APR library
Platform: i686-pc-linux-gnu
checking for working mkdir -p... yes
APR Version: 0.9.3
checking for chosen layout... apr
checking for gcc... gcc
...
```

If `configure` finds any errors or unmet dependencies on your machine, it informs you. Correct the problem and run the `configure` command again until it generates no errors.

The next command, `make`, compiles Apache following the directives in the make files created by the configure step. The output of the `make` command resembles the following code:

```
Making all in srclib
make[1]: Entering directory '/root/temp/httpd-2.0.45/srclib'
Making all in apr
make[2]: Entering directory '/root/temp/httpd-2.0.45/srclib/apr'
Making all in strings
make[3]: Entering directory '/root/temp/httpd-2.0.45/srclib/apr/strings'
make[4]: Entering directory '/root/temp/httpd-2.0.45/srclib/apr/strings'
/bin/sh /root/temp/httpd-2.0.45/srclib/apr/libtool --silent --mode=compile
gcc -g -O2 -pthread   -DHAVE_CONFIG_H -DLINUX=2 -D_REENTRANT -
D_XOPEN_SOURCE=500 -D_BSD_SOURCE -D_SVID_SOURCE -D_GNU_SOURCE   -
I../include -I../include/arch/unix  -c apr_cpystrn.c && touch
apr_cpystrn.lo
...
```

After several minutes, the compilation is complete and you are returned to a system prompt. If any errors are found during the process, they are reported. Fix the problems encountered and rerun the `make` command.

The last command, `make install`, installs Apache into the default location you specified with the `-prefix` parameter. The output of the `make install` command resembles the following:

```
Making install in srclib
make[1]: Entering directory '/root/temp/httpd-2.0.45/srclib'
Making install in apr
make[2]: Entering directory '/root/temp/httpd-2.0.45/srclib/apr'
Making all in strings
make[3]: Entering directory '/root/temp/httpd-2.0.45/srclib/apr/strings'
make[4]: Entering directory '/root/temp/httpd-2.0.45/srclib/apr/strings'
make[4]: Nothing to be done for 'local-all'.
make[4]: Leaving directory '/root/temp/httpd-2.0.45/srclib/apr/strings'
make[3]: Leaving directory '/root/temp/httpd-2.0.45/srclib/apr/strings'
Making all in passwd
make[3]: Entering directory '/root/temp/httpd-2.0.45/srclib/apr/passwd'
...
```

At this point, Apache should be installed into the directory you specified in the first step. To verify that Apache is installed, run the server with the `-v` parameter. This does not start the server but causes it to output its version information. The server executable is named `httpd` (HTTP Daemon) and is located in the `bin` subdirectory of the Apache install directory. Running `httpd` with the `-v` switch should output something similar to the following:

```
Server version: Apache/2.0.45
Server built:   Apr  8 2003 01:42:52
```

If you do not see this information the server possibly did not compile or install correctly. Verify that the httpd executable exists and try running it again, giving the complete path on the command line (for example, `/usr/local/apache2/bin/httpd -v`).

To start the server, use the Apache control script, `apachectl`, as follows:

```
/usr/local/apache2/bin/apachectl start
```

To verify that the server is running, you can use the process command, `ps`, as follows:

```
ps -A | grep "httpd"
 8094 ?        00:00:00 httpd
 8095 ?        00:00:00 httpd
 8096 ?        00:00:00 httpd
 8097 ?        00:00:00 httpd
 8098 ?        00:00:00 httpd
 8099 ?        00:00:00 httpd
```

You should see several instances of httpd running; the actual number depends on the server's configuration.

Installing Apache on Linux from Packages

Most Linux distributions have their own packaging scheme, a means of distributing programs so that they can be easily installed onto systems. Red Hat distributions, for example, have the *RPM* (*R*ed Hat *P*ackage *M*anager) format. The advantages to using packages are as follows:

- The programs can typically be located and installed very easily. For example, you can easily find packaged programs on the Red Hat network, and you can download and install them by using the Red Hat Update Agent.
- Packages typically handle dependency issues for you. That is, if you need other tools or programs to use a particular program, the package has those dependencies encoded in it and at least alerts you before you try to use the new program.
- The packages expand themselves, installing their components into the correct directories — typically with one command. Applications that are installed from packages also follow the base distribution's conventions for locations of binaries, configuration files, and so on.

As previously mentioned, in the section "Linux Downloads," Red Hat 8.0 ships with a version of Apache version 2. Using one of the built-in file browsers, such as Nautilus, as shown in Figure 1-7, you can install an RPM simply by double-clicking its name.

You can also use the console `rpm` command to install a package. The form of the command for RPM installation is as follows:

```
rpm -i <rpm file name>
```

To install the Apache RPM that ships with Red Hat 8.0, for example, you would use the following command:

```
rpm -i httpd-2.0.40-8.i386.rpm
```

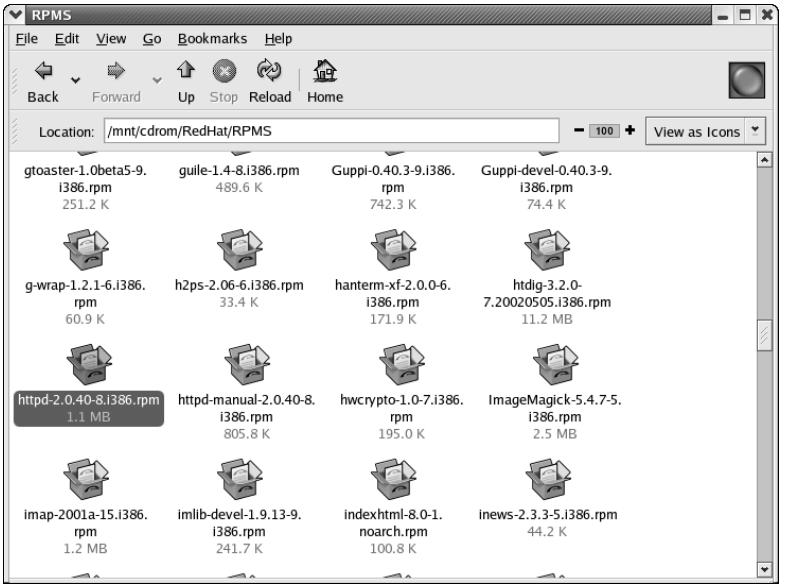


Figure 1-7 *You can install a package by finding it with a file manager (such as Nautilus, pictured) and double-clicking the file name.*

The rpm application inspects the package, tests the system for any dependencies, and installs the package. Notice that, if you use the RPM package to install Apache, the various pieces are installed according to Red Hat's file location scheme. The binary is placed in `/usr/sbin`, the configuration files in `/etc/httpd`, and so on. The rest of this book assumes that you installed Apache on a Red Hat system, using the Red Hat Apache RPM.

If you do not install the latest version of Apache from Apache.org, using your distribution's update service to see whether any critical updates have been made to Apache since the version you installed was released is a good idea. If you are using Red Hat, you can use the Red Hat Update Agent, as shown in Figure 1-8, to download and install any updates.

You can test whether the server was installed by running it with the `-v` parameter, as follows:

```
/usr/sbin/httpd -v
```

The server should respond with its version and build date.

Start the server by using the `apachectl` script with the following command:

```
/usr/sbin/apachectl start
```

Figure 1-8 *You can use the Red Hat Update Agent to update the packages on your system with the latest and greatest available from Red Hat.*

10 Min. To Go

Testing the Installation

After Apache is installed and running, you can test it simply by pointing a browser at the machine running the server. On the server machine itself, you can point a browser to the following address:

```
http://localhost
```

A page similar to the one shown in Figure 1-9 should appear. If you are using another machine to connect to the server, replace `localhost` with the server's fully qualified name or its IP address.

An unmonitored Web server can present a security hazard to the system running it, as well as to any attached network(s). Immediately implementing security measures or stopping the server whenever you don't need it is wise. Rudimentary security measures are covered in Session 4. To stop the server, use the following command:

```
/usr/sbin/apachectl stop
```

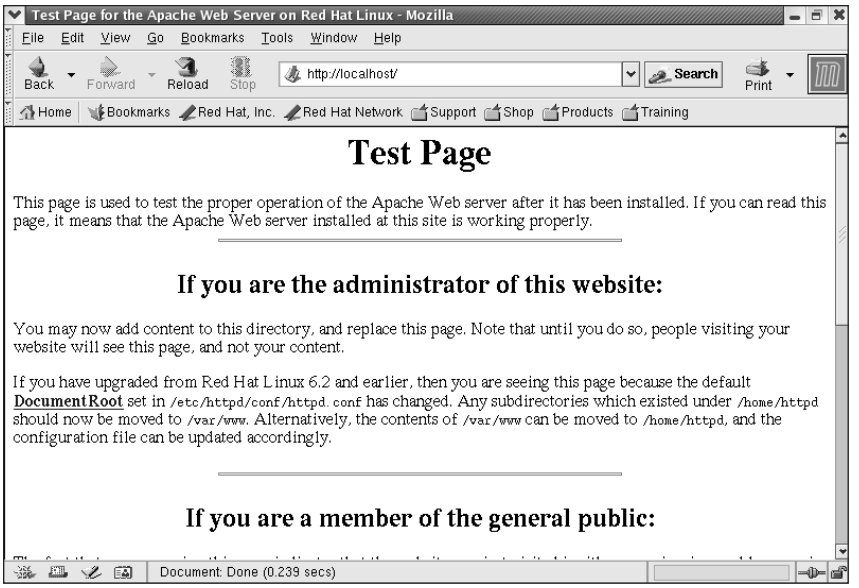


Figure 1-9 *The Apache test page confirms that the server is up and running.*

Done!

REVIEW

This session showed you how to determine what you need to install the Apache HTTP server. You learned where to find the components you need, how to compile the Linux-based server from source code, and how to install the server on both Windows and Linux. Finally, you learned how to start and stop the server, as well as how to verify that it is running.

QUIZ YOURSELF

1. What organization is responsible for the Apache HTTP Project? (See "Apache Is Continually Undergoing Rapid Development.")
2. Why would you want to compile Apache from source files? (See "Apache Is Open Source.")
3. What is the Windows application for controlling Apache called, and how do you access it? (See "Installing Apache on Windows.")
4. What is the Linux application for controlling Apache called, and how do you access it? (See "Building and Installing Apache for Linux (from Source)" or "Installing Apache on Linux from Packages.")