DEVELOPING WITH LAMP

ApacheCon Europe 2005

Daniel Lopez Ridruejo
daniel@bitrock.com

1. Introduction

The LAMP acronym was first coined by Michael Kunze in an article for the German computing magazine c't in 1998 (December Issue, page 230,) to give a name to the set of open source software programs gaining popularity for use together to run dynamic web pages and servers. As you may know, LAMP is an acronym for Linux, Apache, MySQL and PHP (although Python or Perl is often used in its place.) There are several similar platforms that substitute one or more of the open source components of LAMP to serve the same purpose. Examples of this include LAPP, which substitutes PostgreSQL for MySQL, and WAMP, which substitutes Windows for Linux. Regardless of the exact combination of freely available software used, these platforms provide a robust, open source alternative to proprietary web application development environments, such as Microsoft's ASP.NET. Of course, for this tutorial, we will focus on the "standard" LAMP components: Linux, Apache, MySQL, and PHP.

Apache

Back in the early 1990's, the NCSA Web server software was extremely popular. Because many of the original developers of the server left their work to join Netscape, development slowed and eventually halted. The NCSA server was open source, and users of the software started exchanging software patches to fix bugs and improve the functionality of the Web server. Ten years ago last February, they got together and created The Apache Group, a group of developers working on "a patchy server," hence the name. It has since become the most popular web server on the Internet with over 47 million websites running Apache, which represents a 69.6% market share, according to Netcraft's July 2005 survey. Apache now powers many of the busiest Web sites in the world, such as Amazon.com and itunes.com (more notable examples are included below in the discussion of LAMP as a whole.) Obviously, these Web sites have strict requirements for uptime and scalability, and Apache is able to live up to their high standards.

In 1999, The Apache Group incorporated as the Apache Software Foundation (ASF), a
not-for-profit corporation. From the Apache Web site:

> *The Apache Software Foundation will provide organizational, legal, and financial support for the Apache open-source software projects. The Foundation ensures the continuity of Apache projects beyond the participation of individual volunteers, enables contributions of intellectual property and financial support on a sound basis, and provides a vehicle for limiting legal exposure while participating in open-source projects.*

The Apache Software Foundation provides a common umbrella for a variety of Apache
and Web-related technologies.

The Apache license allows for both commercial and noncommercial distribution, modification, and usage. The code is developed by a large, distributed group of talented developers. The open source nature of Apache allows for many other advantages, such as improved stability, security, and customization. Because Apache code is available, and Apache is used widely, the server has been continuously improved and tuned over the years. Apache 2.0 can be configured as a threaded server, a process-based server, or a mixture of the two. This allows the administrator to balance the performance and stability needs in a particular setup. Process-based servers are stable but do not scale as well as threaded servers. Well-designed threaded servers can be faster but less stable if a thread misbehaves.

The open source nature of Apache makes it possible for an attacker to analyze the code, searching for possible vulnerabilities or denial-of-services attacks. Fortunately, it also allows developers to do the same. Changes to the source code are watched for insecurities and are subject to extensive peer review. This encourages high coding standards, and security issues are detected before they go into the code and become problems. No serious remote vulnerabilities have been discovered in Apache for years. When problems arise in Apache or one of its modules, fixes are available within hours or days.

Apache is extremely flexible in terms of operating systems, extensibility, and development languages. It runs on nearly every flavor of Unix and Windows, and other operating systems such as OS/390 and BeOS. This allows enterprises and service providers to standardize on a common Web serving platform across a heterogeneous collection of machines, operating systems, and application servers. It also has a powerful modular API that allows the server to be extended in a variety of ways. Apache 2.0 allows developers to create their own protocol handlers (like FTP or POP3), thus allowing Apache to become a general server framework. Other modules offer template frameworks, authentication, XML processing, and interfaces to enterprise data sources such as directories and relational databases. Apache works with a variety of development languages, including C, C++, Perl, Python, PHP, Tcl, and Java. It provides a shared, common framework between the languages.

Administrative costs are an important part of any software solution. The scarcer certain knowledge and skill sets are, the more expensive it is to hire people with that expertise.
The popularity of Apache means that many Web masters and system administrators are familiar with it. In addition, Apache is commonly used in university projects and research because source code is available. This makes it easy to hire experienced developers and administrators familiar with Apache.

MySQL

MySQL is the most popular open source database available, with over 6 million active installations. Recent polls indicate that MySQL is the third most popular database in use today. MySQL is used throughout the world by Fortune 100 companies, governments, and small businesses alike. Examples of MySQL deployments include Sabre Holdings, Nokia, Lucent, T-Systems, Hewlett Packard, the Associated Press, the French Ministry of Defense, and CNET Networks.

The open source nature of MySQL's code provides for several benefits. In fact, a December 2003 study conducted by Reasoning found that the code quality of MySQL was six times better than that of comparable proprietary code. As with Apache, PHP, and other open source applications, this is because a huge community of developers tests the software across a range of platforms before it is certified for production. Bugs are found and fixed quickly. Access to source code ensures a thorough understanding of the system. Developers can also make modifications or performance enhancements as necessary.

PHP

Ten years ago, Rasmus Lerdorf started development on Personal Home Page Tools. His goal was to provide a better tool for embedding dynamic content into HTML pages. Now known as PHP, it is considered by many to be the ideal tool for web development. Much of the server side scripting language's syntax is based on C, Java, and Perl, which allows people without a strong programming background to quickly learn the language. Another considerable advantage of PHP is its flexibility. Being open source and cross-platform provides for a great deal of freedom for developers, who can easily port code between operating systems, databases and Web servers. PHP will run in most flavors of Unix and Windows, as well as Mac OS X, OS/2 and others. It also supports Open Database Connectivity (ODBC) and therefore offers connectivity to a wide range of common databases, including MySQL, PostgreSQL, Sybase, MySQL, and Microsoft SQL Server. PHP will also easily integrate with web servers aside from Apache, including Microsoft's IIS, Netscape, and less popular servers like thttpd and AOLserver.This allows you to standardize on a common development language across a heterogeneous environment of systems and servers. You can build a solution with PHP on a specific platform/server/database combination, and then migrate to a different combination gradually, replacing one component at a time. You can develop your code on a Windows workstation running IIS and deploy it on a Unix server running Apache with few or no changes.

As of July 2005, 21,466,638 servers connected to the Internet were using PHP, which represents about 30% of the total (according to Netcraft.) With it's popularity, it is not surprising that PHP has a large and very active community supporting it. They have created a large number external libraries and code samples so that you can quickly put together advanced application that do everything from generating Flash and PDF documents to parsing XML.

The LAMP Platform

LAMP has become one of the platform of choice for both casual and high-performance web development. Three out of the top four most reliable company hosting sites for the month of June were using Linux and Apache, and two of those were using PHP as well (the third did not specify a language.) Notable examples of LAMP deployments include: NASA, Wikipedia, the US Census Bureau, and Hoovers.

Now that we have established that LAMP is an extremely popular platform for web development, let's take a look at why that is the case. All of the components share some important characteristics: they are open source, work across multiple platforms, and they can be used free of charge (as long as the licenses are complied with.)

Open Source

I have mentioned some of the primary benefits of open source software throughout this introduction. Because it is one of the key reasons for its success, we will quickly review these important advantages in relation to the LAMP platform:

- Access to the Code: Having access to the code allows anyone to make improvements and modifications as they see fit. This creates a wide knowledge, developer and user-base, which in turn improves the performance, security, and general quality of the code.

- Lower Cost of Ownership: Aside from being available without license fees, a huge amount of information for administering and trouble-shooting open source is available on the Internet. Also, the large number of open source developers means that the cost of finding an hiring someone with expertise is much lower than with highly-specialized, proprietary solutions.

- Freedom from Vendor Lock-In: Apache, MySQL, and PHP will all run across multiple platforms. This means that code can be ported between hardware and operating systems with minimal changes. They will all also integrate with a wide range of both open source and proprietary software and tools. This freedom makes them all extremely attractive to proprietary solutions.

Additional Advantages

In addition to the advantages of being open source, PHP and MySQL are relatively easy to learn and use, so people without previous programming experience can quickly be “up and running” with both. The pluggable architectures of PHP and Apache make it possible for people to write plug-ins and modules to allow users to add additional functionality the server. This creates an ecosystem of modules, which makes the platform much more attractive (chances are that someone has already developed a plug-in for the feature you require.) These significant advantages clearly explain why LAMP has become the “standard” for web deployment.

About This Tutorial

This tutorial will provide a basic introduction to LAMP, along with covering how to build, configure, and administer a LAMP environment. I will also take you through creating a basic application, as well as discussing some cool modules that can enhance LAMP. In essence, the tutorial will be an overview and a basic “how-to” for LAMP. In the interest of working within the scope and time constraints of the session, this tutorial will not serve as a comprehensive, in-depth review of LAMP or any of the components.

2. Installing and customizing Apache

The first step toward setting your LAMP environment is to download, compile, and install a basic Apache server on Linux. This section also covers binary installations.

Choosing the Appropriate Installation Method

Several options are available to get a basic Apache installation in place. Apache is open source, meaning that you can have access to the full source code of the software, which in turn enables you to build your own custom server.

Additionally, pre-built Apache binary distributions are available for most modern Unix platforms. Finally, Apache comes already bundled with a variety of Linux distributions, and commercial versions can be purchased from software vendors.

Building from Source

Building from source gives you the greatest flexibility, enabling you to build a custom server, remove modules you do not need, and extend the server with third-party modules. Building Apache from source code enables you to easily upgrade to the latest versions and quickly apply security patches. Updated versions from vendors usually take days or weeks to appear.

Building Apache from the source code is not that difficult for simple installations, but can grow in complexity when third-party modules and libraries are involved.

Installing a Binary

Unix binary installations are available from vendors and can also be downloaded from the Apache Software Foundation Web site. They provide a convenient way to install Apache for first-time users.

Third-party commercial vendors provide prepackaged Apache installations together with an application server, additional modules, support, and so on.

Installing Apache on Unix

This section explains how to install Apache on Unix and Unix-like systems such as Linux.

Checking Whether Apache Is Already Installed in Your System

If you are running a modern Linux distribution, chances are that Apache is already installed in your system. Try the following at the command-line prompt:

```
# httpd -v
```

Because some distributions name the Apache binary httpd2, you can also try the following:

```
# httpd2 -v
```

If Apache is installed and the binary is in your path, you will get a message with the version and build time:

```
Server version: Apache/2.0.28
Server built: Dec 29 2001 10:32:01
```

Otherwise, you will get command not found or a similar message. It might be that Apache is already installed but is in a different path or with a different binary name, such as `httpd2`. Check whether `/usr/local/apache2/` or `/etc/httpd2` exists and contains a valid Apache 2.0 installation.

An existing 1.3 Apache installation is likely to interfere with your new Apache if the older installation runs at startup, so make sure that either the package is removed from the operating system or the startup script, if any, is disabled. For example, in most Linux distributions, this means modifying the startup scripts at `/etc/rc.d/`.

Installing from Source

The steps necessary to successfully install Apache from source are:

1. Downloading the software
2. Running the configuration script
3. Compiling the code and installing it

These steps are described now in detail.

Downloading the Apache Source Code

The official Apache download site is located at

`http://www.apache.org/dist/httpd`

You can find several Apache versions, packaged with different compression methods. The distribution files are first packed with the Unix tar utility and then compressed either with the `gzip` tool or the compress utility. Download the `.tar.gz` version if you have the gunzip utility installed in your system. This tool comes installed by default in most Linux distributions.

The file you want to download will be named something similar to `httpd-2.0.54.tar.gz` if you are going to use Apache 2.0 or something similar to `apache_1.3.33.tar.gz` if you are going to use Apache 1.3. The rest of this document assumes you use Apache 2.0

Uncompressing the Source Code

You can uncompress the file you just downloaded using the gunzip utility (part of the `gzip` distribution).

You can uncompress and unpack the software by typing the following command:

```
# gunzip < httpd-2.0.54.tar.gz | tar xvf -
```

Uncompressing the file creates a structure of directories, with the top-level directory named `httpd-2.0.54`. Change your current directory to the top-level directory.

Configuring the Software

You can specify which features the resulting binary will have by using the configure script, in the top-level distribution directory. By default, Apache will be compiled with a set of standard modules compiled statically and will be installed in the `/usr/local/apache2` directory. If you are happy with these settings, you can issue the following command to configure Apache:

```
# ./configure
```

However, if you want to be able to extend the server later with third-party modules, such as PHP, you should compile Apache with loadable module support. This, combined with the Apache extension utility (apxs), will enable you to extend the server later with third-party modules without the need to recompile.

To configure Apache this way, issue the following command:

```
#./configure --enable-so --enable-mods-shared=most
```

If you are installing Apache as a normal user and you don't have write permissions on `/usr/local/`, or you simply want to install Apache on a different location, you can specify an alternative location using the `--prefix option`. For example, the following line:

```
#./configure --enable-so --enable-mods-shared=most
-- prefix=/home/username/apache2
```

will compile Apache to be installed in the home directory of the username user.

The purpose of the configure script is to figure out everything related to finding libraries, compile time options, platform-specific differences, and so on, and to create a set of special files called makefiles.

Makefiles contain instructions to perform different tasks, called targets, such as building Apache. These files will then be read by the Unix make utility, which will carry on those tasks. If everything goes well, after executing configure, you will see a set of messages related to the different checks just performed and you will be ready to compile the software.

Compiling and Installing Apache

The make utility reads the information stored in the makefiles and builds the server and modules. Type make at the command line to build Apache.

```
#./make
```

You will see several messages indicating the compilation progress. After compilation is finished, you can install Apache by typing make install.

```
#./make install
```

The Apache distribution files will be copied to `/usr/local/apache2` or the target directory specified with the --prefix switch.

Apache Compilation Options

The Apache configuration script, configure, can take additional options. Many of them are irrelevant for most users, either because they are rarely used or they relate to building Apache distribution packages. A number of them deal with enabling or disabling specific Modules. Below you will find a description of the most useful configuration options. You can get a complete listing by issuing the `./configure --help` command.

Configuration Options

`--with-mpm=mpm` Specifies the Apache Multi-Processing Module. If this option is not specified, the default MPM for the platform will be compiled in. In Unix, the value for mpm can be either worker, perchild, or prefork.

`--enable-so` Enables loadable module support.

`--prefix=path` Apache will be installed relative to the value of the path directory. By default, Apache will be installed in `/usr/local/apache2`.

`--enable-module` Enables or disables the specified module.

`--disable-module` Complete module listing and descriptions.

`--enable-modules=list` Another way of specifying which modules to build, either

`--enable-mods-shared=list` Compiled into the server or as shared libraries. Both switches can take either a list of modules, all (all modules bundled with Apache), or most (includes the majority of the modules you will need).

Selecting the Appropriate MPM

MPMs are available only in Apache 2.0. There are two main MPMs: prefork that will be used by default, and worker. We will use prefork because a threaded server such as worker causes problems with non-thread-safe PHP extensions

Installing Binaries

This section explains how to install a pre-built Apache server on Unix platforms.

Binaries from the Apache Web Site

You can download binaries for different platforms from the Apache Web site at `http://www.apache.org/dist/httpd/binaries`. Check whether binaries for your platform are available. You can download and uncompress the tarball as described in the previous section. In this case, the configuration and compilation steps are not necessary. You can install the software by executing the install-bindist script. You can pass an optional argument, the target installation directory. Otherwise, the software will be installed in `/usr/local/apache2`.

Directory Structure

The four more relevant directories of an Apache default installation are the `conf/` directory, the `bin/` directory, the `htdocs/` directory and the `logs/` directory.

Configuration Files

Apache is configured using directives. Apache keeps its configuration in text files. The main file is called httpd.conf. Additionally you can use other configuration files using the Include directive. Changes to the configuration files do not take effect until you restart the server.

Htdocs Directory

The directory `htdocs` of the Apache default installation contains the documents that will be accessible through the Web site. This directory is also called the document root. If you already have a web page and you want to serve its content with Apache, you can do so simply by copying your files to the this directory.

Apache Binary and Support Scripts

Under bin/ You will find the Apache program executable called, httpd, as well as additional support scripts to easily start and stop the server, manipulate password files, and perform benchmarking and log file processing in the bin directory of the Apache default installation.

Starting Apache

Although you can start Apache by invoking the httpd binary final, it's recommended to used the apachectl control script. The difference is that this script sets certain environment variables and then invokes the `httpd` binary file. After being invoked, the first thing that httpd does is to locate and read the configuration file `httpd.conf`. You can start Apache from the command line by issuing:

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

If no error is found, you will see a message similar to:

```
/usr/local/apache2/bin/apachectl start: httpd started
```

Testing your installation

This indicates the server is up and running. You can test so by opening a browser and accessing the following URL `http://127.0.0.1/`, which will take you to the test page. You may need to use `http://127.0.0.1:8080/` if you are installing as a regular user and using an unprivileged port.

Stopping Apache

You can stop Apache from the command line issuing:

```
# /usr/local/apache2/bin/apachectl stop
```

After a moment you should see a message similar to:

```
/usr/local/apache2/bin/apachectl stop: httpd stopped
```

Basic Apache Configuration

Once Apache starts, it will create two log files, the access_log and the error_log. You can find both files at the logs directory of the Apache default installation.

The `access_log` file is used to track client requests. When a client requests a document from the server, Apache records several parameters associated with the request in this file, such as: the IP address of the client, the document requested, the HTTP status code, and the current time. You can change the name and location of the `access_log` editing the `httpd.conf` file and modifying the value specified in the `CustomLog` directive.

The `error_log` file is used to record important events. This file includes error messages, startup messages, and any other significant events in the life cycle of the server. This is the first place to look when you run into a problem when using Apache. You can change the name and location of the `error_log` editing the httpd.conf file and modifying the value specified in the `ErrorLog` directive.

With the default configuration, Apache will wait for requests in the port 80. You can change that by editing the `httpd.conf` file and modifiying the value specified in the `Listen` directive.

The `ServerName` directive sets the hostname and port that the server uses to identify itself. This is used when creating redirection URLs. The `ServerName` directive accepts a fully qualified domain name and an optional port. If the port is not specified, it is assumed to be the port from the incoming request. For example, you can use:

```
ServerName www.example.com:80
```

If no `ServerName` is specified, then the server attempts to deduce the `hostname` by performing a reverse lookup on the IP address of the TCP connection. If no port is specified in the `ServerName`, then the server will use the port from the incoming request. For optimal reliability and predictability, you should specify an explicit hostname and port using the ServerName directive.

You can find more information about Apache in the technical documentation that is located in the `/usr/local/apache2/htdocs/manual` directory.

3. Installing and customizing PHP

Installing on Unix

This section explains you how to install PHP 4 on Unix as an Apache 2 module.

Binary Installation

Most Linux and Unix distributions that come preinstalled with Apache already include a PHP module. Use your package manager to find out whether the PHP package is already installed in your system; otherwise, download and install it. You can also check your distribution Web site and download the package from there.

Source Code Installation

This section explains how to download, build, and install the PHP module.

Getting PHP

The PHP source code can be downloaded from the main PHP Web site at `http://www.php.net`. The file you need is called `php-4.3.11.tar.gz`. After you have downloaded the tarball, uncompress it and change to the newly created directory:

```
# gunzip < php-4.3.11.tar.gz | tar xvf -
# cd php-4.3.11
```

The PHP directory structure contains the following important directories, as shown in

`TSRM/` The Thread Safe Resource Manager.
`Zend/` The code for the Zend scripting engine.
`build/` The build-related scripts and Makefiles.
`ext/` The extensions bundled with PHP for database access, XML manipulation, and so on. Each of the subdirectories contains a Makefile, and most of them have a README file that explains the purpose of the extension.
`libs/` The directory where the PHP Apache module shared library will be placed when it is built.
`main/` The core of the PHP code.
`modules/` The directory where additional modules shared libraries will be placed.
`pear/` The PEAR (PHP Extension and Application Repository) contains a collection of reusable library code similar to Perl's CPAN (http://www.cpan.org).
`regex/` The regular expression library code.

`sapi/` The server extension abstraction layer. Here you can find modules to interface PHP to Microsoft IIS, Netscape, and, of course, Apache.
`scripts/` The miscellaneous scripts used by PHP developers.
`rests/` The test suite.

Compiling PHP

PHP, like many other open source projects, uses autoconf and automake tools to ease portability. The build scripts are able to find out by themselves most of the information they need to compile PHP, but you must pass certain parameters explicitly. PHP will be built as a loadable module, with the help of the Apache `apxs` tool.

The rest of this section assumes that you have installed Apache 2.0 in `/usr/local/apache2` and that you have root privileges. Apache must have been compiled with loadable module support enabled (`--enable-so` option). PHP will be installed under `/usr/local/php4`. If you want to build Apache and PHP as a regular user, you must change the paths provided later to paths that you have write permissions to. Type the following in the directory created when you uncompressed the PHP 4 sources:

```
./configure --with-apxs2=/usr/local/apache2/bin/apxs --with-mysql --
prefix=/usr/local/php4
```

You will see a rapid succession of messages while the configure script checks for the libraries it needs in your system and creates the Makefiles necessary for the build system. If everything goes well and the configure script finishes without throwing any errors, you can type the following to build PHP:

```
make
```

After the build finishes, you will have a `libphp4.so` file in the `libs/` directory.
To install the files, type

```
make install
```

This will perform the following tasks:
• Installs the shared library `libphp4.so` into the `/usr/local/apache2/modules` directory
• Adds a LoadModule directive into `/usr/local/apache2/conf/httpd.conf`
• Installs PHP header files, binaries, and the PEAR libraries into `/usr/local/php4`

Finally you will have to edit your `/usr/local/apache2/conf/httpd.conf` and add the following line:

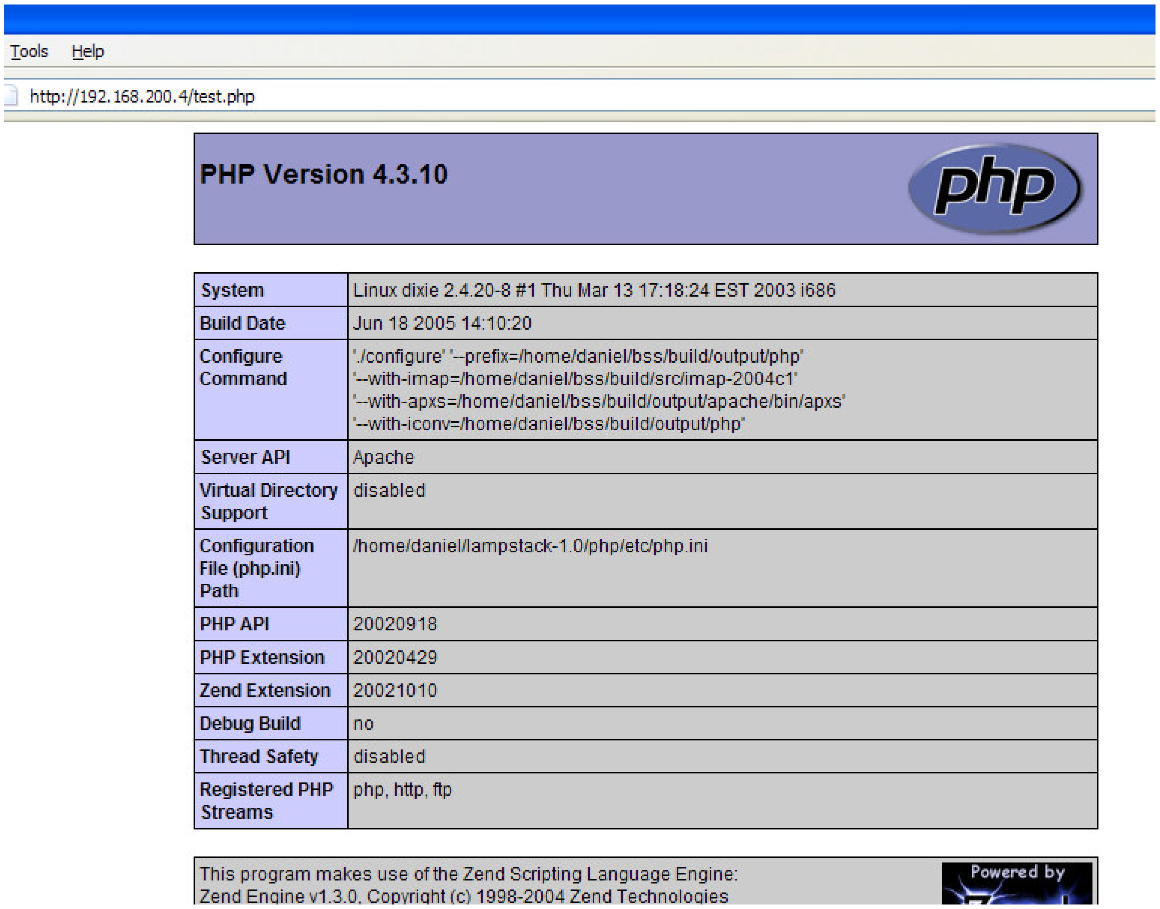
```
AddType application/x-httpd-php .php
```

Testing Your PHP Installation

To test whether PHP is working correctly, create a file called `example.php` in the `/usr/local/apache2/htdocs` directory with the following contents:

```
<?php phpinfo(); ?>
```

Restart Apache and access the URL `http://127.0.0.1/example.php`. If PHP was installed correctly, you should see a something similar to this:



PHP Version 4.3.10

System	Linux dixie 2.4.20-8 #1 Thu Mar 13 17:18:24 EST 2003 i686
Build Date	Jun 18 2005 14:10:20
Configure Command	'./configure' '--prefix=/home/daniel/bss/build/output/php' '--with-imap=/home/daniel/bss/build/src/imap-2004c1' '--with-apxs=/home/daniel/bss/build/output/apache/bin/apxs' '--with-iconv=/home/daniel/bss/build/output/php'
Server API	Apache
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/home/daniel/lampstack-1.0/php/etc/php.ini
PHP API	20020918
PHP Extension	20020429
Zend Extension	20021010
Debug Build	no
Thread Safety	disabled
Registered PHP Streams	php, http, ftp

This program makes use of the Zend Scripting Language Engine:
Zend Engine v1.3.0, Copyright (c) 1998-2004 Zend Technologies

The Apache error file `/usr/local/apache2/logs/error_log` is the first place to look if you get an error or empty page. It will provide you with valuable information about what might have gone wrong. By far the most common issue is permissions: Make sure that the file `example.php` is readable by the user which Apache is running as.

Also make sure that Apache was built with loadable module support. You can check that by issuing the following command:

```
/usr/local/apache2/bin/httpd -l
```

and looking for `mod_so.c` in the output. Make sure that PHP is being loaded by checking for the appropriate `LoadModule` directive in `httpd.conf`.

In this section you learned how to build PHP as an Apache module and how to configure it to work with your Apache server. That was a basic installation. The PHP configure command allows many more flags to be passed to enable different language features and extensions. You can get a complete list of the options by typing:

```
./configure --help | more
```

The most important ones are:

General Options

The following are general options that you can pass to the configure script.

```
--prefix=/some/path
```

Allows you to specify the path where PHP will be installed.

```
--with-apxs2=/path/to/apache2/bin/apxs
```

Builds shared Apache 2.0 module using the specified apxs utility.

```
--enable-debug
```

Enables debug symbols; useful for troubleshooting.

```
--without-pear
```

Does not install PEAR.

```
--enable-safe-mode
```

Enables the restricted safe mode by default. If you use this option, you will also be interested in `--with-exec-dir`, which specifies the executables allowed in safe mode.

```
--with-openssl
```

Includes OpenSSL support. OpenSSL is a library that provides SSL support

```
--with-curl
```

Includes curl support. Libcurl is a library that provides client-side support for a variety of protocols, including HTTP/HTTPS, FTP, Telnet, and more. You can learn more about curl at `http://curl.sourceforge.net/`.

```
--enable-ftp
```

Enables FTP support.

Graphics Support

The following commands are options that you can pass to the configure script to configure PHP graphics-related libraries.

```
--with-gd=/path/to/gd/install/dir
```

GD is a library that allows programmatic image creation and manipulation. It is useful for generating on-the-fly images and logos. You can learn more about GD at `http://www.boutell.com/gd/`. The previous command line will build GD as part of PHP. If you want to create a shared library, you must pass the command line as

```
--with-gd=shared,/path/to/gd/install/dir
```

GD depends on additional libraries to support certain graphic formats. Associated configure options are

```
--with-jpeg-dir=/path/to/jpeg/install/dir
```

For libjpeg support.

```
--with-png-dir=/path/to/libpng/install/dir
```

For libpng support.

```
--with-xpm-dir=/path/to/libxpm/install/dir
```

For libXpm support.

```
--with-t1lib=/path/to/t1lib/install/dir
```

For t1lib, Adobe Type 1 fonts support.

GD allows the use of TTF (TrueType Fonts) to add text to images:

```
--enable-gd-native-ttf
```

Enables TrueType string function in GD.

```
--with-freetype-dir=/path/to/freetype2/install/dir
```

FreeType 2 support.

```
--with-ttf=/path/to/freetype/install/dir
```

Includes FreeType 1.x support.

An additional PHP module provides improved graphic manipulation using the imlib graphics library. You can find more information at `http://mmcc.cx/php_imlib/`.

Flash Animation

PHP provides Shockwave Flash support via two libraries: SWF and Ming. To install the SWF library:

```
--with-swf=/path/to/swf/install/dir
```

The SWF library can be found at `http://reality.sgi.com/grafica/flash/`.

To install the Ming library:

```
--with-ming=/path/to/ming/install/dir
```

The Ming library provides support for Flash generation and includes a PHP binding. It can be found at `http://opaque.net/ming`.

PDF Generation

PHP supports on-the-fly generation of PDF documents using the `clibpdf` and `pdflib` libraries:

```
--with-pdflib=/path/to/pdflib/install/dir
```

PDF support via the pdflib library requires a license for commercial usage. You can learn more about pdflib at `http://www.pdflib.com/pdflib/index.html`.

```
--with-cpdf=/path/to/clibpdf/install/dir
```

PDF generation support via the clibpdf library. You can learn more at `http://www.fastio.com/`.

Database Support

PHP supports a variety of database backends.

```
--with-mysql=/path/to/mysql/dir
```

Support for the MySQL (`http://www.mysql.com`) database. If the path is not specified, PHP includes built-in support and will use it instead.

```
--with-pgsql=/path/to/pgsql/dir
```

Support for the PostgreSQL database (`http://www.posgresql.org`).

XML Support

The following are options that you can pass to the configure script to configure PHP's XML-related libraries.

```
--with-dom=/path/to/libxml/install/dir
```

Includes DOM support via the `libxml` library, a C-based XML processing library distributed under the LGPL and the W3C IPR licenses. You can learn more about libxml at `http://xmlsoft.org/`.

```
--disable-xml
```

Disables built-in expat XML support (it is on by default).

```
--enable-xslt
```

Enables XSLT support.

```
--with-sablot=/path/to/sablotron/install/dir
```

Provides support for the Sablotron XSLT transformation engine. You can learn more about Sablotron at `http://www.gingerall.com/`.

```
--with-expat-dir=/path/to/expat/install/dir
```

Expat library required by Sablotron. You can find expat at `http://www.jclark.com/xml/expat.html`.

```
--with-qtdom
```

XML DOM support via the qt library that can be found at `http://www.trolltech.com/products/qt/`.

```
--enable-wddx
```

Enables wddx support, which is used when programming Web services.

Session Support

The following are options that you can pass to the configure script to configure PHP session support. This enables PHP scripts to keep track of user data between requests.

```
--enable-trans-id
```

Enables transparent ID propagation of session information (this can be done via cookies).

```
--with-mm
```

Enables shared memory support for session storage via the mm library. You can learn more about the mm library at `http://www.engelschall.com/sw/mm/`.
This section presented you with several configuration options to give you an idea of the capabilities of PHP. There are many more that provide support for additional databases, SNMP, CORBA, calendar functions, IMAP, Unicode, Java, LDAP, encryption, and more. You can get a comprehensive description of supported language features at `http://www.php.net/manual/en/`.

You can find additional extensions and PHP Web applications in Freshmeat at `http://freshmeat.net` and in the SourceForge PHP foundry at `http://sourceforge.net/foundry/php-foundry/`. For example, the Vagrant charting extension `http://vagrant.sourceforge.net` provides support for programmatic generation of graphic charts.

Examples of Web applications based on PHP are Phorum (`http://phorum.org/`) for Web discussion boards and IMP (`http://www.horde.org/imp/`) for Web mail. Nuke (`http://phpnuke.org/`) and Midgard (`http://www.midgard-project.org/`) are content management/Web portal systems.

PHP Configuration

PHP can be configured either via the `php.ini` file located in `/usr/local/php4/lib/` or from inside the Apache configuration file. You can copy the file `php.ini-dist` from the build directory to `/usr/local/php4/lib/php.ini`. The `php.ini` consists of key/value pairs. The same settings can be specified in the Apache configuration file with the use of these directives:

```
php_value name value
```

Sets the value of the name variable to value.

```
php_flag name on|off
```

Sets a Boolean configuration option.

There are certain options, called admin options, that must be specified in the main Apache configuration file. They can be set using `php_admin_value` and `php_admin_flag`. These options are usually security related, such as `open_basedir` or `safe_mode_exec_dir`.

Some of the configuration options are relevant to PHP and others are for configuring specific PHP modules. The following is a selection of the available configuration options.

PHP Language

You can modify the way PHP can be mixed with HTML tags with the following options.

```
short_open_tag boolean
```

To include PHP code, you usually need to surround it with `<?php` or `<script>` tags. The `short_open_tag` directive enables you to use `<? ?>` tags in your code, although PEAR coding practices encourage you to use the `<?php` format.

asp_tags boolean Allows use of ASP-style tags `<% %>` and constructs `(<%=$varname %>` to include the value of a variable.

```
memory_limit integer
max_execution_time integer
```

These two directives set the maximum amount of memory in bytes that a script is allowed to allocate, and the maximum time in seconds that a script is allowed to run before the script is terminated by the PHP engine, respectively. This helps to protect server resources from poorly written scripts.

```
include_path string
```

Specifies a list of directories where certain PHP functions (for including other files and so on) look for files.

Error Manipulation

```
display_errors boolean
```

Determines whether errors should be printed to the screen as part of the HTML output.

```
error_log string
```

Specifies the name of the file to which script errors should be logged. If the special value syslog is used, the errors are sent to the Unix system logger instead.

Output Manipulation

Apache transmits to the network the content created by the PHP script as it is being generated. You might want to add specific headers to a response, but are unable to do so because you have already sent part of the content. If you enable output buffering, PHP will cache the page, enabling you to set headers at any point on the page. PHP also provides hooks so that the content generated can be filtered or changed. As an example, PHP supports compression of the output of a script if the browser can understand compressed content, thus minimizing download time. PHP also provides the ability to append or prepend headers or footers to all generated pages, thus easing the task of creating a consistent, sitewide look and feel.

```
auto_append_file string
auto_prepend_file string
```

PHP makes it possible to append or prepend files to every page served. These files are parsed and interpreted as PHP scripts. If the name of the file is none, auto-prepending or appending is disabled.

```
output_buffering boolean
```

Enables or disables output buffering.

```
output_handler handler
```

Allows the specification of an output handler, such as `ob_gzhandler` for compression.

Security

It is possible to configure PHP to enhance the security of the installation, especially in environments with multiple or not fully trusted users. PHP allows a safe mode operation, which restricts the PHP/system functionality that the scripts can access, such as limiting access to only certain files or directories. It is possible to configure PHP to run as a CGI. This has advantages and risks from a security standpoint, such as the ability to use the Apache suexec wrapper. Many of the security issues need to be handled or complemented at the PHP level with safe coding practices. You can learn more at `http://www.php.net/manual/en/security.php`.

```
safe_mode boolean
```

Specifies whether to enable PHP's safe mode.

```
safe_mode_exec_dir string
```

Specifies that system calls executing external programs will work only with binaries in this directory.

```
open_basedir string
```

If present, this directive limits the files that can be opened by PHP to the ones contained under the specified directory path.

Dynamic Extension Support

You can either compile PHP extensions into the PHP executable, or you can choose to compile the extensions themselves as shared objects and load them from within the PHP engine.

```
enable_dl boolean
```

Enabled by default, this directive restricts the ability to load shared library code into PHP. The main reason to disable dynamic loading is security. Dynamic loading is not available when using PHP in safe mode.

```
extension_dir string
```

Specifies the directory in which PHP should look for dynamically loadable extensions.

```
extension string
```

Specifies which dynamically loadable extensions to load when PHP starts.

PHP Basics

This section gives you a quick overview of PHP basic syntax, so you get a feel for the language. The actual programming and operation is explained later in the tutorial.

As well as in MySQL, all PHP statements must finish with a semicolon, also known as the instruction terminator.

Comments can be specified in a variety of ways, such as /* */ and //

Variables: A variable in PHP consists of a name of your election, preceded by a dollar sign. Variable names can include letters, numbers and the underscore character, must begin with a letter or an underscore, but cannot include spaces. You can assign a value to a variable using the following syntax:

```
$var_name = value;
```

In PHP there are three types of variables: local to a function or script, global within a function or script and superglobals.

Data Types: In PHP there are eight data types.

- Integer: A whole number
- Double: A floating-point number
- String: A collection of characters
- Boolean: One of the special values true or false
- Object: An instance of a class
- Array: An ordered set of keys and values
- Resource: Reference to a third-party resource
- NULL: An uninitialized variable

Constants: PHP allows you to create constants using the built-in define() function. The basic syntax is:

```
define( "CONSTANT_NAME", value);
```

The value you want to set can be a number, a string, or a boolean. By convention the name of the constant should be in capital letters. Constants are accessed with the constant name only; no dollar symbol is required.

Flow control functions syntax

IF

```
if ( expression ) {
   // code to execute if the expression evaluates to true
} else {
   // code to execute in all other cases
}

elseif

if ( expression ) {
   // code to execute if the expression evaluates to true
} elseif (another expression) {
   // code to execute if the previous expression failed
   // and this one evaluates to true
} else {
   // code to execute in all other cases
}
```

Switch

```
switch ( expression ) {
       case result1:
            // execute this if expression results in result1
            break;
       case result2:
            // execute this if expression results in result2
            break;
       default:
            // execute this if no break statement
            // has been encountered hitherto
}
```

Basic loop syntax

The `while` Statement

```
while ( expression ) {
      // do something
}
```

The `do...while` Statement

```
do   {
     // code to be executed
} while ( expression );
```

The `for` Statement

```
for ( initialization expression; test expression; modification
expression ) {
    // code to be executed
}
```

Functions

You can define a function using the `function` statement:

```
function some_function( $argument1, $argument2 ) {
     // function code here
}
```

Calling a function:

```
Function_name($arg1, $arg2);
```

You can return values from functions using a `return` statement.

4. Installing MySQL

How to download and install MySQL
Basic security guidelines for running MySQL

Choosing the Appropriate Installation Method

The easiest way to install MySQL in Linux is using a rpm distribution. If your Linux distribution is not rpm based, you can also use a pre-packaged binary distribution. This last method requires having the gunzip and tar tools present in your system which are installed by default in Linux and you will need to have enough privileges to create users and groups in the target system.

Either way, you should secure the MySQL installation at the end of the process.

Downloading the software

The official MySQL download site is located at `http://dev.mysql.com/downloads/` You can find several MySQL versions. We will use MySQL 4.1.12 standard version which is the latest generally available recommended release at the time this tutorial was written.

Installing using a rpm distribution

If you choose to install from rpm, the option that is most likely to fit your needs is the one under the section Linux x86 RPM downloads.

For a minimal installation you will need to download two files. First, the MySQL server whose name should be `MySQL-server-4.1.13.i386.rpm`, and secondly, the standard MySQL client libraries, `MySQL-client-4.1.13.i386.rpm`. You will find both files under the Linux x86 RPM downloads section.

Once you have downloaded the files issue the following command:

```
# rpm -i MySQL-server-4.1.13.i386.rpm  MySQL-client-4.1.13.i386.rpm
```

Layout of a rpm installation:

`/usr/bin` Client programs and scripts
`/usr/sbin` The mysqld server
`/var/lib/mysql` Log files, databases
`/usr/share/doc/packages` Documentation
`/usr/include/mysql` Include (header) files
`/usr/lib/mysql` Libraries

`/usr/share/mysql`	Error message and character set files
`/usr/share/sql-bench`	Benchmarks

Installing from a binary distribution

The steps necessary to successfully install MySQL from a binary distribution are:

1. Downloading the software
2. Uncompressing the software.
2. Execute post-installation procedures

These steps are described now in detail.

Downloading the MySQL binary distribution

As we previously stated, the official MySQL download site is located at `<http://dev.mysql.com/downloads/>`, where you can find several MySQL versions. We will use MySQL 4.1.12 standard version which is the latest Generally Available recommended release.

The option that is most likely to fit your needs among binary distibutions is the one under the section Linux downloads (x86, glibc-2.2, static, gcc).

The file you have to download will have a name similar to `mysql-standard-4.1.12-pc-linux-gnu-i686.tar.gz.`

Uncompressing the binary distribution

You can uncompress the file you just downloaded using the gunzip utility (part of the gzip distribution).

Change your current directory to `/usr/local` issuing:

```
# cd /usr/local
```

You can now uncompress and unpack the software by typing the following command:

```
# gunzip < mysql-standard-4.1.12-pc-linux-gnu-i686.tar.gz | tar xvf -
```

Uncompressing the file creates a structure of directories, with the top-level directory named `mysql-standard-4.1.12-pc-linux-gnu-i686`. Change the name of the the top-level directory to msql:

```
# mv mysql-standard-4.1.12-pc-linux-gnu-i686 mysql
```

Layout of a MySQL binary installation

`bin` Client programs and the mysqld server
`data` Log files, databases
`docs` Documentation, ChangeLog
`include` Include (header) files
`lib` Libraries
`scripts` mysql_install_db
`share/mysql` Error message files
`sql-bench` Benchmarks

Post-Installation Procedures

Add a login user and group for mysql:

```
# /usr/sbin/groupadd mysql
# /usr/sbin/useradd -g mysql mysql
```

Create the MySQL grant tables:

```
# cd /usr/local/mysql
# scripts/mysql_install_db --user=mysql
```

Change the ownership of program binaries to root and the ownership of the data directory to the user that you just created to run mysql.

```
# cd /usr/local/mysql
# chown -R root   .
# chown -R mysql data
# chgrp -R mysql .
```

Starting MySQL

After installing MySQL you can start the MySQL server issuing:

```
$ /usr/local/mysql/bin/mysqld_safe --user=mysql &
```

You can verify that MySQL is up and running using mysqladmin, for example, issuing:

```
$ /usr/local/mysql/bin/mysqladmin version
```

Testing MySQL

Once the MySQL server has been started you can test that you can retrieve information from the server using the following examples:

```
$ /usr/local/mysql/bin/mysqlshow
```

```
$ /usr/local/mysql/bin/mysqlshow -u root mysql
```

Stopping MySQL

You can shutdown MySQL with the following command:

```
$ /usr/local/mysql/bin/mysqladmin -u root shutdown
```

Securing Your Installation

The MySQL grant tables define the initial MySQL user accounts and their access privileges. The default configuration consists of:

- Two privileged accounts with a username of 'root' for connection from the local host. These accounts have initially no password. That means that anyone will be able to connect to the MySQL server as a superuser without being asked for a password, and will be granted all privileges. To assign passwords to the root accounts, make sure MySQL is started and use the following commands:

```
   $ mysql -u root

   mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('newpwd');
   mysql> SELECT Host, User FROM mysql.user;
```

  Replace `host_name` in the following command with the result you got from executing the command above:

```
   mysql> SET PASSWORD FOR 'root'@'host_name' = PASSWORD('newpwd');
```

- Two anonymous-user accounts, both with an empty username for connection from the local host. None of these accounts have a password initially. That means that anyone will be able to connect to the MySQL server as a regular user without being asked for a password. These anonymous accounts have all privileges for the test database or other databases with names that start with `test_`

  To assign passwords to the anonymous account, make sure MySQL is started and issue:

```
   $ mysql -u root
   mysql> SET PASSWORD FOR ''@'localhost' = PASSWORD('newpwd');
   mysql> SELECT Host, User FROM mysql.user;
```

  Replace `host_name` in the following command with the result you got from executing the command above:

```
   mysql> SET PASSWORD FOR ''@'host_name' = PASSWORD('newpwd');
```

It is strongly recommended that you do not have empty passwords for any user accounts before using the server for any production work.

MySQL Basics

Connecting and disconnecting from the server

In order to connect to a MySQL server, you usually have to invoke the `mysql` binary providing a MySQL username and if you specified a password for that user, a password as well. If you are trying to connect to a remote MySQL server, you will also need to provide a hostname. The command will look similar to this:

```
$ mysql -h hostname -u user -p
Enter password: ********
```

If that works, some introductory information followed by a `mysql>` prompt will show up on the screen:

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 4.1.12-standard
```

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

```
mysql>
```

The prompt tells you that the server is ready to receive commands.

The default MySQL installation allows users to connect to the server on the local host using anonymous accounts. In that case, you will be able to connect to that server by invoking `mysql` without any options:

```
shell> mysql
```

You can disconnect from the server at any time by typing QUIT (or \q) or by pressing Contro-D:

```
mysql> QUIT
Bye
```

MySQL Basic Query Commands

MySQL commands are not case sensitive, which means that you can use insert with the same meaning of INSERT or WHERE with the same meaning of WhEre. All MySQL instructions, except exit, have to finish with a semicolon ;

```
INSERT:
```

Once you have created a table, the INSERT SQL command is used for adding new records to it.

The basic syntax of INSERT is:

```
INSERT INTO table_name (column list) VALUES (column values);
```

SELECT: SELECT is the MySQL command used to retrieve records from tables. The syntax of this command can be from very simple to really complicated depending on which fields from a table you want to select. The most basic SELECT syntax looks like:

```
SELECT expressions_and_columns FROM table_name
[WHERE some_condition_is_true]
[ORDER BY some_column [ASC | DESC]]
[LIMIT offset, rows]
```

WHERE: This SQL command is used to specify a particular condition. Its syntax is:

```
WHERE condition_is_true
```

ORDER: This command allows you to order the result of SELECT queries in a specific way. Syntax:

```
SELECT column_1, column_2, column_3 from table_name ORDER BY column_2;
```

LIMIT: The LIMIT command allows you to return only a certain number of records from a SELECT query result. Basic syntax:

```
SELECT column_1, column_2, column_3 from table_name ORDER BY column_2
       -> LIMIT 2;
```

UPDATE: UPDATE is the SQL command used to modify the contents of one or more columns in an existing record. The most basic UPDATE syntax looks like this:

```
UPDATE table_name
SET column1='new value',
column2='new value2'
[WHERE some_condition_is_true]
```

REPLACE: This command is a different method for modifying records, which is similar to the INSERT command.

```
REPLACE INTO table_name (column list) VALUES (column values);
```

DELETE: The basic DELETE syntax is

```
DELETE FROM table_name
[WHERE some_condition_is_true]
[LIMIT rows]
```

5. Creating your first LAMP application

So far, we have learnt how to install and configure the different components of the LAMP stack. We have learnt about how MySQL works and how to perform basic database-related tasks. We have seen the basic building blocks and syntax of the PHP language.

It is time now to learn how to write our own software applications on top of LAMP. As you will see, PHP and MySQL make it incredibly easy to create simple web applications.

This is possible because both are projects with a focus on ease of use and solving real world problems. You can also take advantage of a great number of open source libraries and components that cover most common aspects of web development. The open nature of the LAMP platform allow your applications to be easily be extended over time to meet new requirements.

The following sections cover the design of a LAMP-based application. For this purpose, our project will be to add a news section to a current static website. We will use the MySQL database to store the news and PHP to integrate them into our website. Having the news items stored in a database will allow us to query, search and manipulate them a variety of ways.

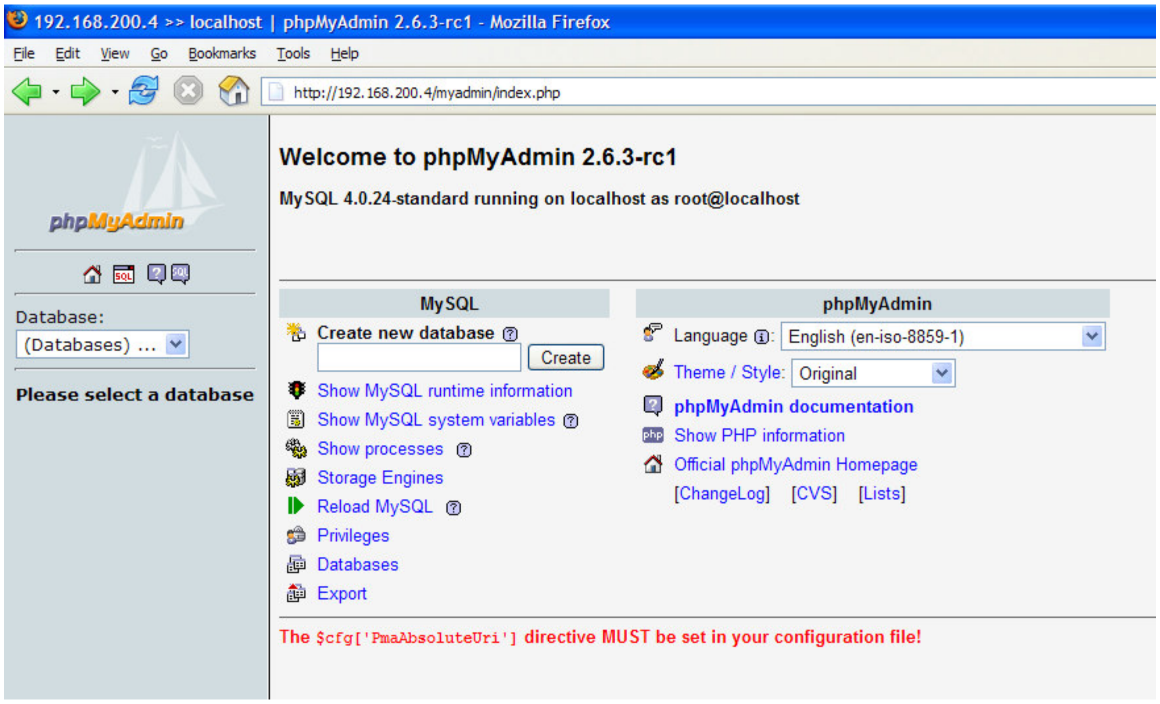
Creating the database

Lets start with the design of the database. In this case we start with a simple table that contains the following columns:

`id` : A unique identifier for our news
`date` : The date for the news item
`title` : The heading for the news
`summary` : A summary for the news
`description` : The full news text

We can create this directly from the command line SQL client, but it is easier to do so using phpMyAdmin, specially when you are just getting started with LAMP. If using the default installation values, installation of phpMyAdmin is as easy as downloading the software from http://www.phpmyadmin.net/ and uncompressing it in the htdocs directory.

We access the phpMyAdmin main page, and create a new database called “lamp” and enter the appropriate definition for the table news


192.168.200.4 >> localhost | phpMyAdmin 2.6.3-rc1 - Mozilla Firefox

http://192.168.200.4/myadmin/index.php

Welcome to phpMyAdmin 2.6.3-rc1

MySQL 4.0.24-standard running on localhost as root@localhost

Database: (Databases) ...

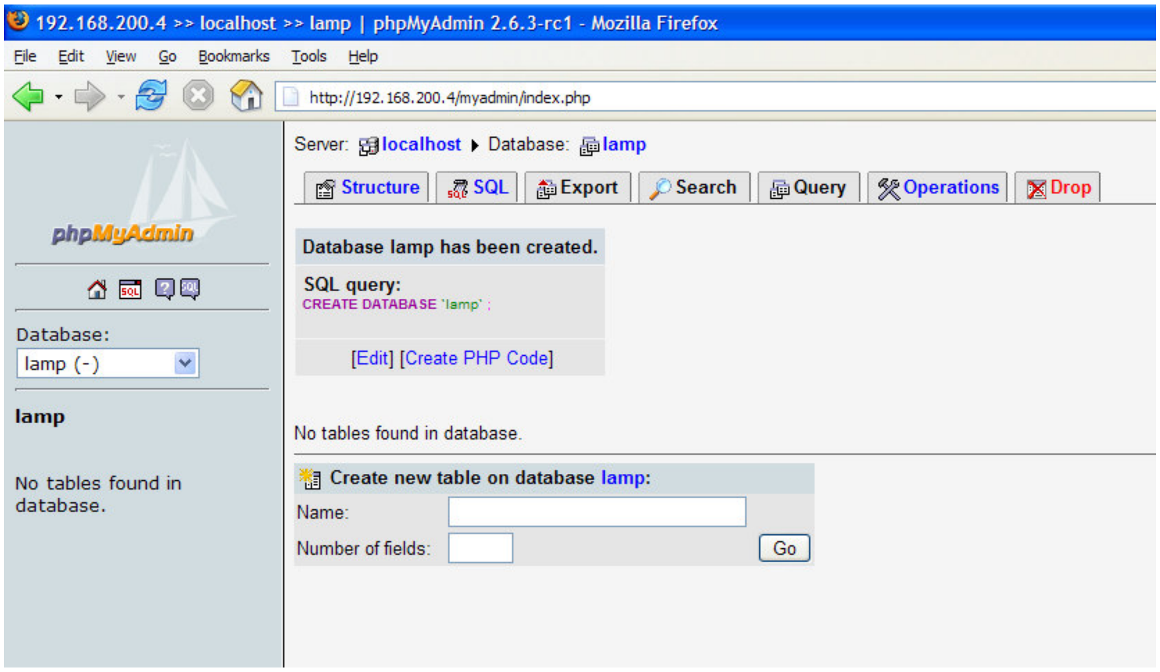
Please select a database

MySQL
- Create new database [Create]
- Show MySQL runtime information
- Show MySQL system variables
- Show processes
- Storage Engines
- Reload MySQL
- Privileges
- Databases
- Export

phpMyAdmin
- Language: English (en-iso-8859-1)
- Theme / Style: Original
- phpMyAdmin documentation
- Show PHP information
- Official phpMyAdmin Homepage
- [ChangeLog] [CVS] [Lists]

The $cfg['PmaAbsoluteUri'] directive MUST be set in your configuration file!

Figure 1 The main phpMyAdmin page


192.168.200.4 >> localhost >> lamp | phpMyAdmin 2.6.3-rc1 - Mozilla Firefox

http://192.168.200.4/myadmin/index.php

Server: localhost ▸ Database: lamp

Structure | SQL | Export | Search | Query | Operations | Drop

Database: lamp (-)

lamp

No tables found in database.

Database lamp has been created.

SQL query:

```
CREATE DATABASE `lamp` ;
```

[Edit] [Create PHP Code]

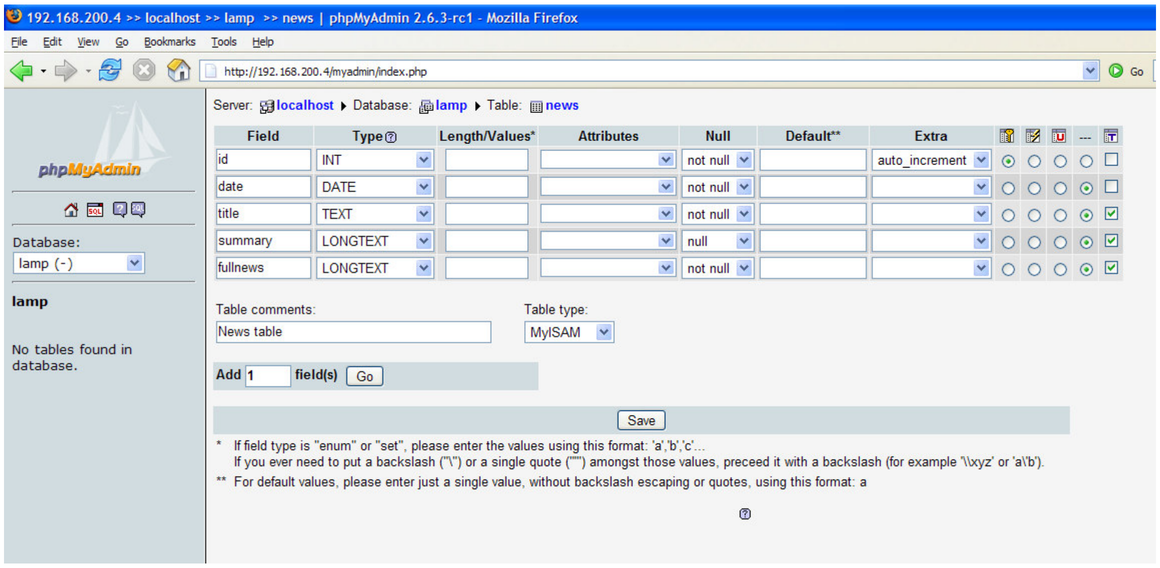
No tables found in database.

Create new table on database lamp:

Name:

Number of fields: [Go]

Figure 2 Creating a MySQL database using phpMyAdmin



Server: localhost ▸ Database: lamp ▸ Table: news

Field	Type	Length/Values*	Attributes	Null	Default**	Extra				---	
id	INT			not null		auto_increment	●	○	○	○	☐
date	DATE			not null			○	○	○	●	☐
title	TEXT			not null			○	○	○	●	✓
summary	LONGTEXT			null			○	○	○	●	✓
fullnews	LONGTEXT			not null			○	○	○	●	✓

Table comments: News table

Table type: MyISAM

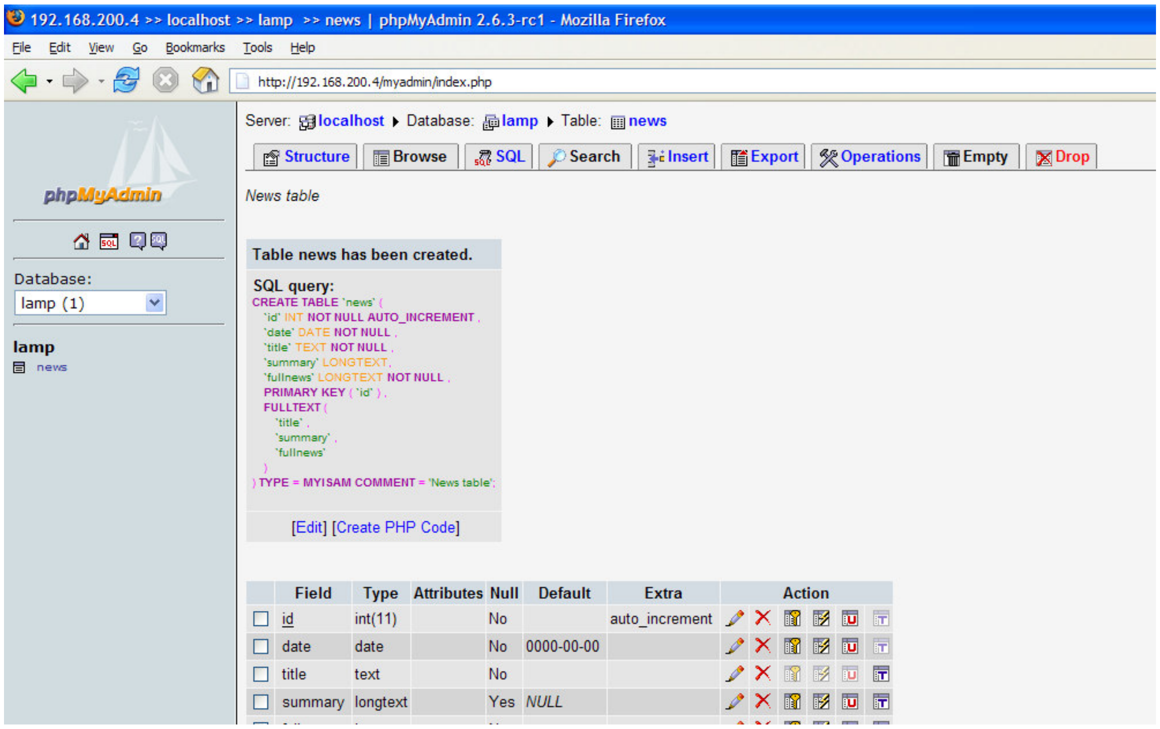
Add 1 field(s)

* If field type is "enum" or "set", please enter the values using this format: 'a','b','c'...
If you ever need to put a backslash ("\") or a single quote ("'") amongst those values, preceed it with a backslash (for example '\\xyz' or 'a\'b').

** For default values, please enter just a single value, without backslash escaping or quotes, using this format: a

Figure 3 Creating a MySQL table using phpMyAdmin

As you can see in the following screenshot, phpMyAdmin will also display the SQL used to create the table. This is very useful for learning and debugging purposes.



Server: localhost ▸ Database: lamp ▸ Table: news

News table

Table news has been created.

SQL query:

```
CREATE TABLE `news` (
  `id` INT NOT NULL AUTO_INCREMENT ,
  `date` DATE NOT NULL ,
  `title` TEXT NOT NULL ,
  `summary` LONGTEXT,
  `fullnews` LONGTEXT NOT NULL ,
  PRIMARY KEY ( `id` ) ,
  FULLTEXT (
    `title` ,
    `summary` ,
    `fullnews`
  )
) TYPE = MYISAM COMMENT = 'News table';
```

[Edit] [Create PHP Code]

	Field	Type	Attributes	Null	Default	Extra	Action
☐	id	int(11)		No		auto_increment	
☐	date	date		No	0000-00-00		
☐	title	text		No			
☐	summary	longtext		Yes	*NULL*		

Figure 4 SQL code used to create the table

Some of the fields in the table have special characteristics:

The "id" field is set to auto_increment. This will allow us to easily have unique ids for each one of our news items. This entry has also been marked as being a primary key. A primary key (or a combination of them) act as a unique identifier for each database entry.

Three fields have been marked as FULLTEXT. This tells MySQL to scan and index their contents, easing the task of searching later on. We will see how this can be accomplished later on in the tutorial.

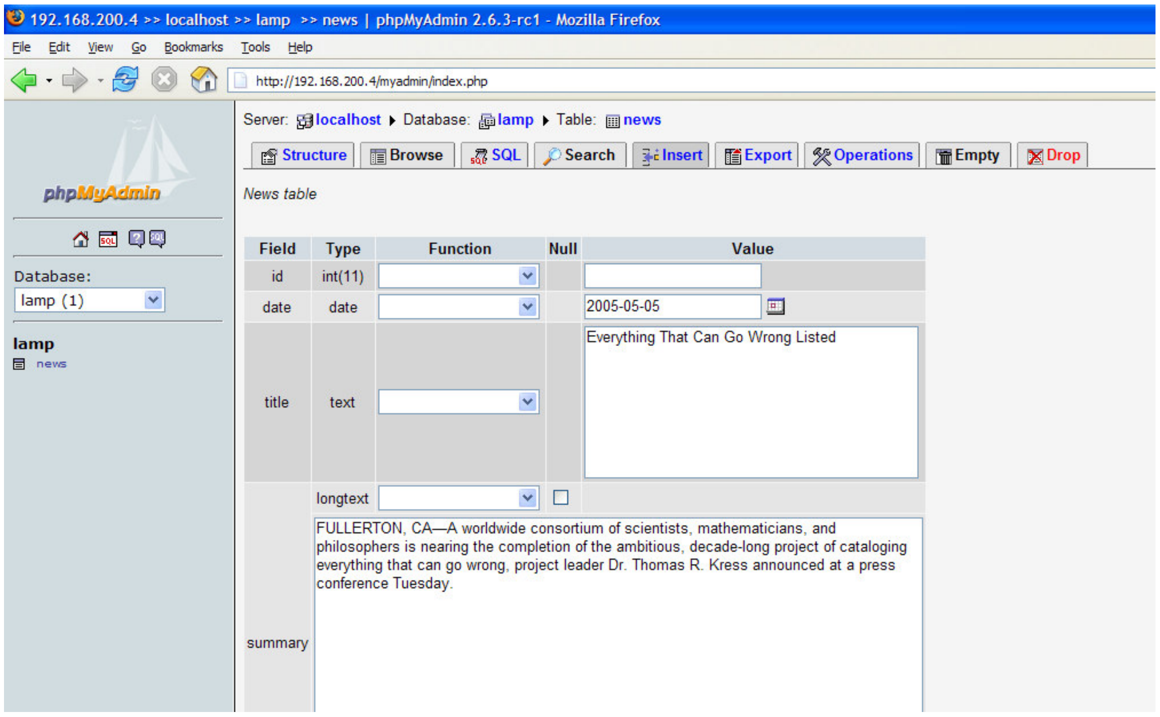
The summary field has been marked so it can contain NULLs, meaning they can be empty. In that case, the news item will only have a title and the main body of the news.

Populating the database

The next step is to add some entries into the database. For now, we will do it using the phpMyAdmin tool. Later on, we will be see how to develop a custom interface.

Since our application is centered around news, we take our examples from "America's finest news source", http://www.onion.com

We will use, once again, the phpMyAdmin tool to populate the database, as shown in figure 5


192.168.200.4 >> localhost >> lamp >> news | phpMyAdmin 2.6.3-rc1 - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://192.168.200.4/myadmin/index.php
phpMyAdmin
Database:
lamp (1)
lamp
news
Server: localhost Database: lamp Table: news
Structure Browse SQL Search Insert Export Operations Empty Drop
News table

Field	Type	Function	Null	Value
id	int(11)			
date	date			2005-05-05
title	text			Everything That Can Go Wrong Listed
summary	longtext		☐	FULLERTON, CA—A worldwide consortium of scientists, mathematicians, and philosophers is nearing the completion of the ambitious, decade-long project of cataloging everything that can go wrong, project leader Dr. Thomas R. Kress announced at a press conference Tuesday.

Figure 5 Creating a MySQL table using phpMyAdmin

Now that we have news in our database, it is time to use PHP to display them in our website. The following code selects all news in our database and displays them in the web page:

```
<style type="text/css"><!-- @import url("lamp.css"); --> </style>
<body>
<?

$conn = mysql_connect("localhost");
mysql_select_db("lamp", $conn);
$sql="SELECT * FROM news";
$result = mysql_query($sql, $conn) or die(mysql_error());

while ($row = mysql_fetch_array($result)) {
    echo "<div class=\"title\">" . $row['title'] . "</div>";
    echo "<div class=\"summary\">" . $row['summary'] . "</div>";
    echo "<div class=\"fullnews\">" . $row['fullnews'] . "</div>";
}

?>
</body>
```

The lamp.css contains a simple cascading style sheet :

```
.title {
      font-size:15px;
      font-weight:bold;
      color:0000cc;
      text-transform:uppercase;
      margin-top:30px;
      width:800px;
      text-align:left;
}

.summary {
      font-size:12px;
      font-weight:bold;
      width:800px;
      text-align:left;
      margin-top: 20px;
      margin-bottom: 20px;
      background:#e5e5e5;
      padding: 10px;
}

.fullnews {
      font-size:12px;
      line-height: 1.5;
      width:800px;
      text-align:left;
      margin-top: 20px;
      margin-bottom: 20px
}
```

Lets analyze in detail what each of the lines does

The mysql_connect function allows you to establish a connection to the database server.

This function takes an argument that is a connection string of the form "hostname:port" or "hostname:/path/to/mysql.sock". The first form describes the hostname or IP address to connect to and the TCP port, in case we are connecting to the database over TCP/IP. Although this carries a performance penalty, it allows you to run your server and database in entirely separate servers. The second form describes connection to the database using local sockets. If not specified in the function call or defined in the php.ini configuration, it defaults to "localhost:3306". A default MySQL install contains allows TCP/IP connections only from the local host, reducing the chances of unauthorized users connecting to the database server

You can also specify the default server to connect to using the `mysql.default_host` configuration setting. Then there is no need to pass an argument to `mysql_connect()`. Note that there is a difference between "localhost" and 127.0.0.1. Whenever you specify "localhost" as the server name, the client library will try to connect to the database using a local socket. If you want to use TCP/IP, you will need to explicitly specify 127.0.0.1

If you need a username/password pair to access the database, you will need to pass those as additional arguments to connect to the database:

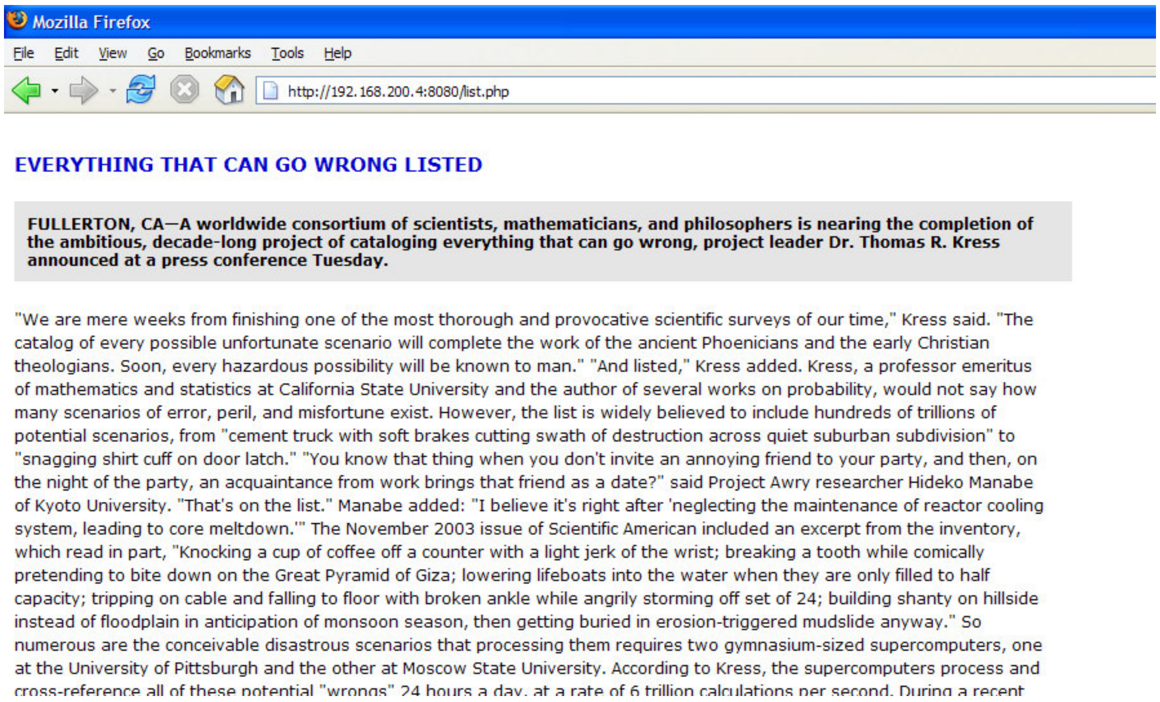
```
$conn = mysql_connect("localhost", “username”, “password”);
```

There are a couple of optional, more advanced options to mysql_connect that are described in detail in the PHP reference manual.

The `mysql_select_db()` function selects a specific database. If the database does not exist, or we do not have permissions to access it, we will get an error. We do not check for the error now, but rather when we execute the query, which is done with `mysql_query()`. The line with `mysql_query()` executes an SQL query in the server and stores the results in the `$result` variable. If there was an error, it will print the related information using `mysql_error()`. The error could be caused because of problems executing the query or because earlier calls to connect to the database did not succeed.

The rest of the code iterates over the returned rows and prints them to the screen. Each one of the rows is retrieved and stored as an array using `mysql_fetch_array`. Each one of the fields in the array can then be accessed using the field names as keys into the associative array.

The following figure shows the result of running the `list.php` script:


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://192.168.200.4:8080/list.php

EVERYTHING THAT CAN GO WRONG LISTED

FULLERTON, CA—A worldwide consortium of scientists, mathematicians, and philosophers is nearing the completion of the ambitious, decade-long project of cataloging everything that can go wrong, project leader Dr. Thomas R. Kress announced at a press conference Tuesday.

"We are mere weeks from finishing one of the most thorough and provocative scientific surveys of our time," Kress said. "The catalog of every possible unfortunate scenario will complete the work of the ancient Phoenicians and the early Christian theologians. Soon, every hazardous possibility will be known to man." "And listed," Kress added. Kress, a professor emeritus of mathematics and statistics at California State University and the author of several works on probability, would not say how many scenarios of error, peril, and misfortune exist. However, the list is widely believed to include hundreds of trillions of potential scenarios, from "cement truck with soft brakes cutting swath of destruction across quiet suburban subdivision" to "snagging shirt cuff on door latch." "You know that thing when you don't invite an annoying friend to your party, and then, on the night of the party, an acquaintance from work brings that friend as a date?" said Project Awry researcher Hideko Manabe of Kyoto University. "That's on the list." Manabe added: "I believe it's right after 'neglecting the maintenance of reactor cooling system, leading to core meltdown.'" The November 2003 issue of Scientific American included an excerpt from the inventory, which read in part, "Knocking a cup of coffee off a counter with a light jerk of the wrist; breaking a tooth while comically pretending to bite down on the Great Pyramid of Giza; lowering lifeboats into the water when they are only filled to half capacity; tripping on cable and falling to floor with broken ankle while angrily storming off set of 24; building shanty on hillside instead of floodplain in anticipation of monsoon season, then getting buried in erosion-triggered mudslide anyway." So numerous are the conceivable disastrous scenarios that processing them requires two gymnasium-sized supercomputers, one at the University of Pittsburgh and the other at Moscow State University. According to Kress, the supercomputers process and cross-reference all of these potential "wrongs" 24 hours a day, at a rate of 6 trillion calculations per second. During a recent

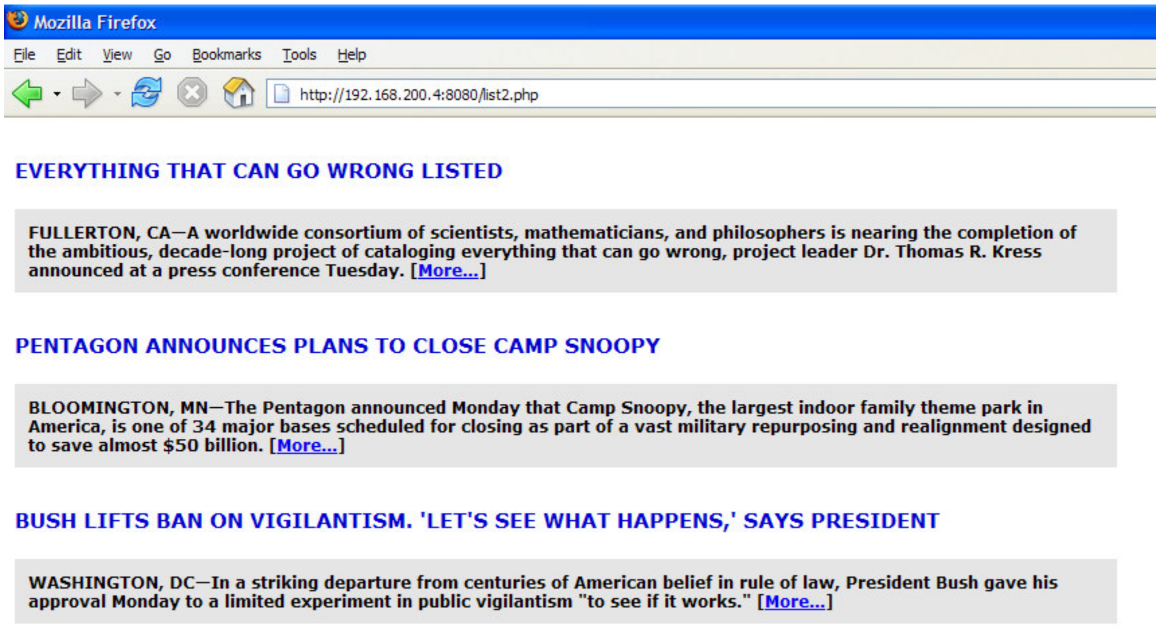
Figure 6 Listing news items

As an alternative, we can display a summary of the news, and generate a link to a different page that displays the full news item, as shown in the following listing:

```
<style type="text/css"><!-- @import url("lamp.css"); --> </style>
<body>
<?

$conn = mysql_connect("localhost");
mysql_select_db("lamp", $conn);
$sql="SELECT * FROM news";
$result = mysql_query($sql, $conn) or die(mysql_error());

while ($row = mysql_fetch_array($result)) {
    echo "<div class=\"title\">" . $row['title'] . "</div>";
    echo "<div class=\"summary\">" . $row['summary'] ;
    echo " [<a href=\"details.php?id=" . $row['id'] . "\">More...</a>]"
. "</div>";
}

?>
</body>
```

Listing : list2.php

Which renders the following:


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://192.168.200.4:8080/list2.php

EVERYTHING THAT CAN GO WRONG LISTED

FULLERTON, CA—A worldwide consortium of scientists, mathematicians, and philosophers is nearing the completion of the ambitious, decade-long project of cataloging everything that can go wrong, project leader Dr. Thomas R. Kress announced at a press conference Tuesday. [More...]

PENTAGON ANNOUNCES PLANS TO CLOSE CAMP SNOOPY

BLOOMINGTON, MN—The Pentagon announced Monday that Camp Snoopy, the largest indoor family theme park in America, is one of 34 major bases scheduled for closing as part of a vast military repurposing and realignment designed to save almost $50 billion. [More...]

BUSH LIFTS BAN ON VIGILANTISM. 'LET'S SEE WHAT HAPPENS,' SAYS PRESIDENT

WASHINGTON, DC—In a striking departure from centuries of American belief in rule of law, President Bush gave his approval Monday to a limited experiment in public vigilantism "to see if it works." [More...]

Figure 7 Listing news items summaries

For each news items, the script generates a URL that includes the ‘id’ value for the item and points to a `details.php` page. This page can then access the value through the _GET global variable, as shown in the next listing:

```
<style type="text/css"><!-- @import url("lamp.css"); --> </style>
<body>
<?

$conn = mysql_connect("localhost");
mysql_select_db("lamp", $conn);
$sql="SELECT * FROM news where id='" . $_GET['id'] . "'";
$result = mysql_query($sql, $conn) or die(mysql_error());

$row = mysql_fetch_array($result);
echo "<div class=\"title\">" . $row['title'] . "</div>";
echo "<div class=\"summary\">" . $row['summary'] . "</div>";
echo "<div class=\"fullnews\">" . $row['fullnews'] . "</div>";

?>
</body>
```
:

Listing : details.php

As you can see, the `details.php` page takes that id, selects the appropriate news item from the database and displays it.

With the help of the previous examples, we already know how to search and display news items. The next section explains how to create, delete and modify items in the database using PHP.

Updating the database

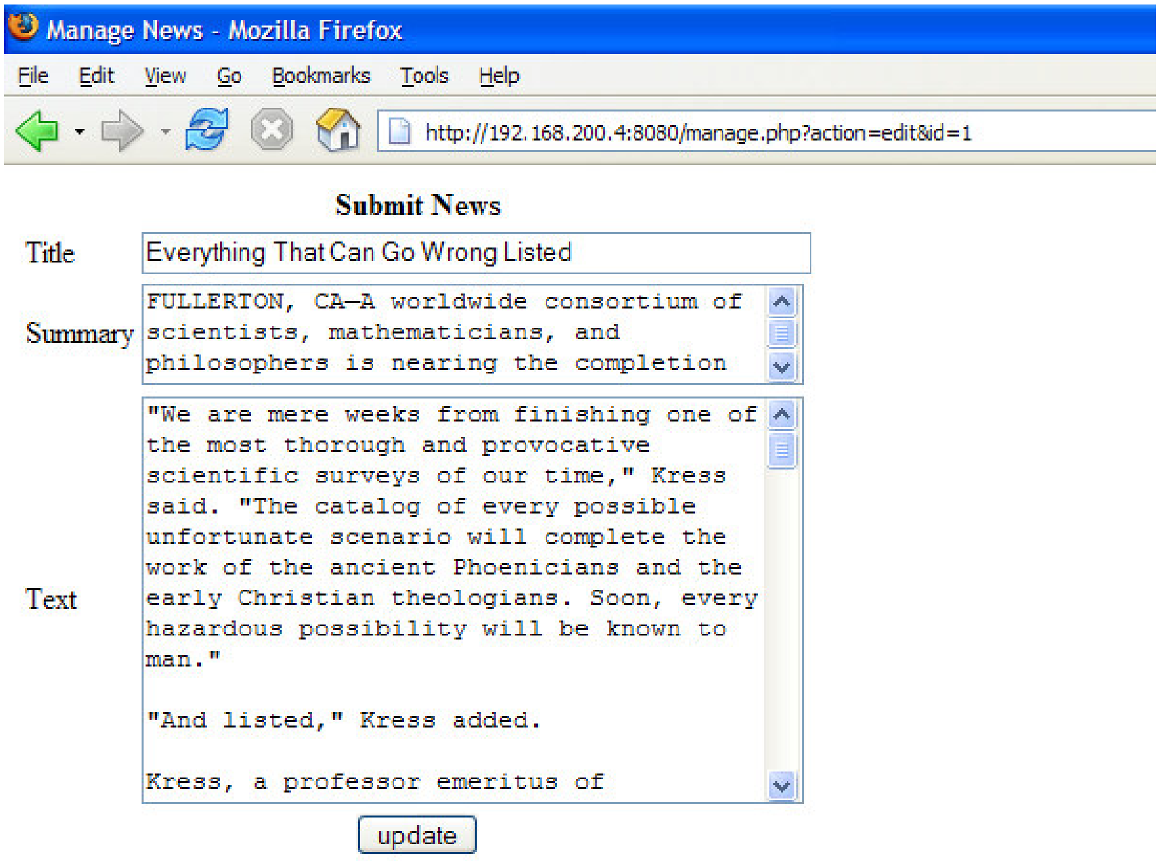
The next step is to create a Web page that provides an easy to use interface so the appropriate user can add new items to the database without having to use a developer tool such as phpMyAdmin.

The first step is to create an HTML form so the user can input the data. We then add the necessary PHP code to handle the form submissions. The end result is shown in the following listing:

```
<html>
<head><title>Manage News</title></head>
<body>

<?
$conn = mysql_connect("localhost");
mysql_select_db("lamp", $conn);

if ($_GET['action'] == "delete") {
    $sql="DELETE FROM news WHERE id ='" .$_GET['id'] . "'";
    $result = mysql_query($sql, $conn) or die(mysql_error());
    echo "News item successfully deleted";
    $action = "insert";
} elseif ($_GET['action'] == "edit") {
    $sql="SELECT * FROM news where id=" .$_GET['id'];
    $result = mysql_query($sql, $conn) or die(mysql_error());
    $row = mysql_fetch_array($result);
    $title = $row['title'];
    $summary = $row['summary'];
    $text = $row['fullnews'];
    $action = "update";
} elseif ($_GET['action'] == "update") {
    $sql = "UPDATE news SET ".
      "title = '" .$_GET['title'] . "'," .
      "summary = '" . $_GET['summary'] . "'," .
      "fullnews = '" .$_GET['fullnews'] ."'" .
      "WHERE id =" .$_GET['id'] ;
    $title = $_GET['title'];
    $summary = $_GET['summary'];
    $text = $_GET['fullnews'];
    $action = "update";
    $result = mysql_query($sql, $conn) or die(mysql_error());
} elseif ($_GET['action'] == "insert") {
    $action = "insert";
    $sql = "INSERT INTO news ( id , date , title , summary , fullnews )
" .
      "VALUES ('', 'now()', '" .
     $_GET['title'] . "', '" .
     $_GET['summary'] . "', '" .
     $_GET['fullnews'] . "');";
    $action = "insert";
    $result = mysql_query($sql, $conn) or die(mysql_error());
} else {
    $action = "insert";
}
```

```
?>
        <form action="manage.php" method="GET"><input type="hidden"
name="id" value="<? echo $_GET['id']; ?>">
        <table>
           <tr>
              <td colspan="2" align="center"><b>Submit News</b></td>
           </tr>
           <tr>
              <td>Title</td><td><input type="text" name="title"
size="54" value="<? echo $title ?>"/></input></td>
           </tr>
           <tr>
              <td>Summary</td><td><textarea name="summary" cols="40"><?
echo $summary ?></textarea></td>
           </tr>
           <tr>
              <td>Text</td><td><textarea name="fullnews" cols="40"
rows="12"><? echo $text ?></textarea></td>
           </tr>
           <tr>
              <td colspan="2" align="center"><input type="submit"
name="action" value="<? echo $action ?>"></td>
           </tr>
        </table>
        </form>

</body>
</html>
```

Listing: manage.php


Manage News - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://192.168.200.4:8080/manage.php?action=edit&id=1
Submit News
update

Title	Everything That Can Go Wrong Listed
Summary	FULLERTON, CA—A worldwide consortium of scientists, mathematicians, and philosophers is nearing the completion
Text	"We are mere weeks from finishing one of the most thorough and provocative scientific surveys of our time," Kress said. "The catalog of every possible unfortunate scenario will complete the work of the ancient Phoenicians and the early Christian theologians. Soon, every hazardous possibility will be known to man." "And listed," Kress added. Kress, a professor emeritus of

Figure 7: Editing a news item

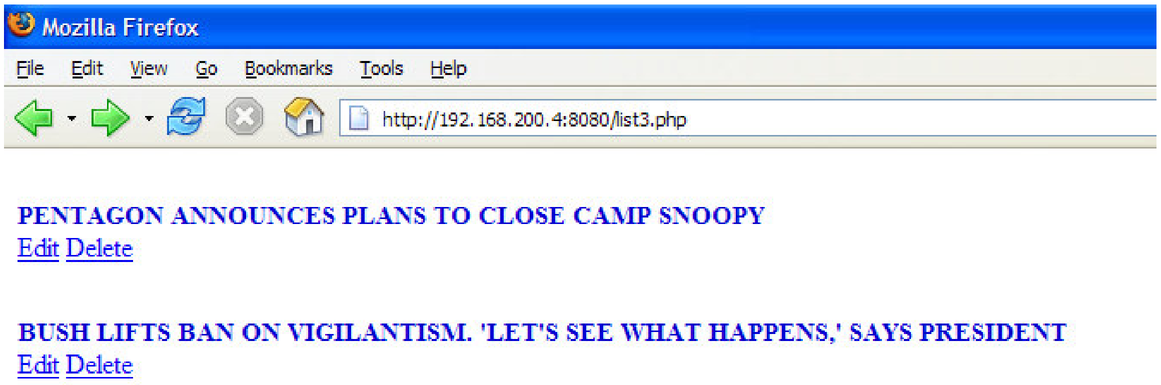
To complement this form, we create a page that list all the news, together with edit, delete buttons next to it that point to `manage.php` with the appropriate action value.

```
<style type="text/css"><!-- @import url("lamp.css"); --> </style>
<body>
<?

$conn = mysql_connect("localhost");
mysql_select_db("lamp", $conn);
$sql="SELECT * FROM news";
$result = mysql_query($sql, $conn) or die(mysql_error());

while ($row = mysql_fetch_array($result)) {
    echo "<div class=\"title\">" . $row['title'] .
      " </div><a href=\"manage.php?action=edit&id=" . $row['id'] .
"\">Edit</a> " .
      " <a href=\"manage.php?action=delete&id=" . $row['id'] .
"\">Delete</a><br>";
}
```

```
?>
</body>
```

Listing: list3.php

Which displays the following:


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://192.168.200.4:8080/list3.php

PENTAGON ANNOUNCES PLANS TO CLOSE CAMP SNOOPY
Edit Delete

BUSH LIFTS BAN ON VIGILANTISM. 'LET'S SEE WHAT HAPPENS,' SAYS PRESIDENT
Edit Delete

Figure 8: Listing news for editing

Searching the database

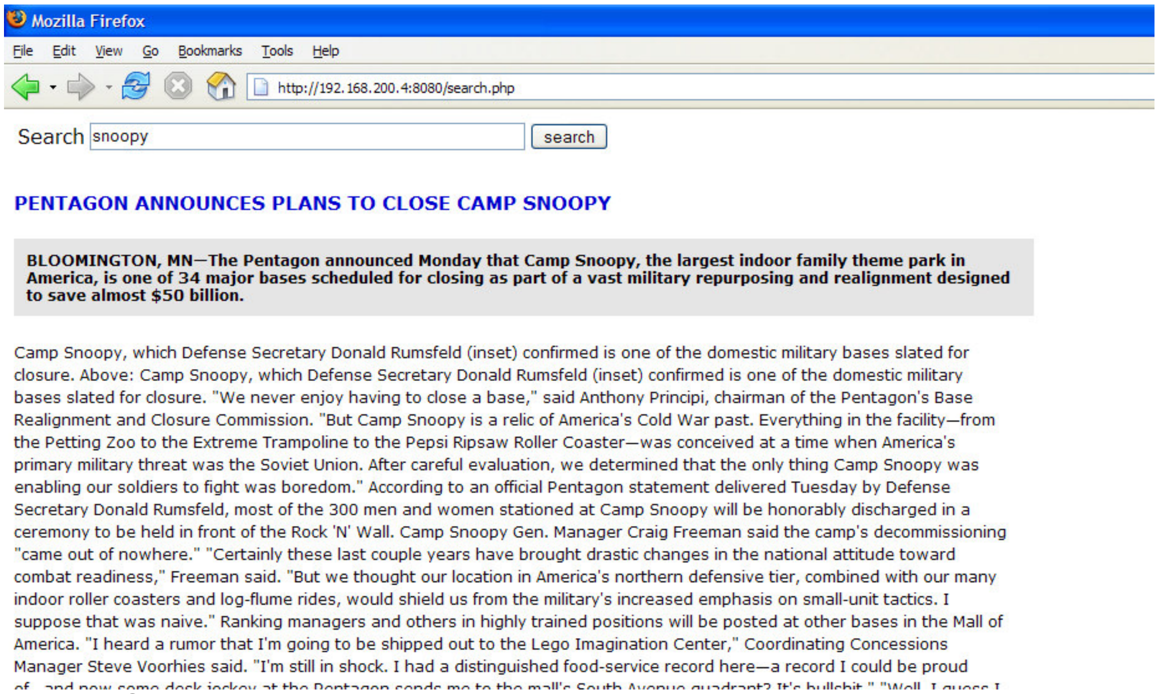
As we mentioned earlier, the FULLTEXT feature of MySQL makes it really easy to add search capabilities to your database, as the following listing demonstrates:

```
<style type="text/css"><!-- @import url("lamp.css"); --> </style>
<body>

        <form action="search.php" method="POST">
            <table>
             <tr>
                <td>Search</td><td><input type="text" name="searchterm"
size="54" value="<? echo $_POST['searchterm'] ?>"/></input></td>
                <td colspan="2" align="center"><input type="submit"
name="search" value="search"></td>
             </tr>
            </table>
         </form>

<?

$conn = mysql_connect("localhost");
mysql_select_db("lamp", $conn);
$sql="SELECT * FROM news WHERE MATCH (title, summary, fullnews) AGAINST
('" . $_POST['searchterm'] . "');";
$result = mysql_query($sql, $conn) or die(mysql_error());

while ($row = mysql_fetch_array($result)) {
    echo "<div class=\"title\">" . $row['title'] . "</div>";
    echo "<div class=\"summary\">" . $row['summary'] . "</div>";
    echo "<div class=\"fullnews\">" . $row['fullnews'] . "</div>";
}
```

```
?>
</body>
```

Listing: search.php

The MySQL database indexes the text in the news items for us and allows us to query easily query them with MATCH.


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://192.168.200.4:8080/search.php
Search snoopy
search

PENTAGON ANNOUNCES PLANS TO CLOSE CAMP SNOOPY

BLOOMINGTON, MN—The Pentagon announced Monday that Camp Snoopy, the largest indoor family theme park in America, is one of 34 major bases scheduled for closing as part of a vast military repurposing and realignment designed to save almost $50 billion.

Camp Snoopy, which Defense Secretary Donald Rumsfeld (inset) confirmed is one of the domestic military bases slated for closure. Above: Camp Snoopy, which Defense Secretary Donald Rumsfeld (inset) confirmed is one of the domestic military bases slated for closure. "We never enjoy having to close a base," said Anthony Principi, chairman of the Pentagon's Base Realignment and Closure Commission. "But Camp Snoopy is a relic of America's Cold War past. Everything in the facility—from the Petting Zoo to the Extreme Trampoline to the Pepsi Ripsaw Roller Coaster—was conceived at a time when America's primary military threat was the Soviet Union. After careful evaluation, we determined that the only thing Camp Snoopy was enabling our soldiers to fight was boredom." According to an official Pentagon statement delivered Tuesday by Defense Secretary Donald Rumsfeld, most of the 300 men and women stationed at Camp Snoopy will be honorably discharged in a ceremony to be held in front of the Rock 'N' Wall. Camp Snoopy Gen. Manager Craig Freeman said the camp's decommissioning "came out of nowhere." "Certainly these last couple years have brought drastic changes in the national attitude toward combat readiness," Freeman said. "But we thought our location in America's northern defensive tier, combined with our many indoor roller coasters and log-flume rides, would shield us from the military's increased emphasis on small-unit tactics. I suppose that was naive." Ranking managers and others in highly trained positions will be posted at other bases in the Mall of America. "I heard a rumor that I'm going to be shipped out to the Lego Imagination Center," Coordinating Concessions Manager Steve Voorhies said. "I'm still in shock. I had a distinguished food-service record here—a record I could be proud of…and now some desk jockey at the Pentagon sends me to the mall's South Avenue quadrant? It's bullshit." "Well, I guess I

Figure 9: Searching

Sending Email

So far we have an interface that allows us to manage the news in our website. Our next task is to add a new feature : We want to receive an email every time a piece of news is added to the database. To do so, we can use the local mail server, if you have one installed, or access a remote mail server using SMTP.

To send email, we use the Mail package. This package is part of the PEAR extension and application repository and it is included by default with your PHP installation. There are other ways of sending email with PHP, but we will not cover them in this tutorial.

The following code allows you to send email to news-alert@example.com :

```
include("Mail.php");
function mail_news($title, $summary, $fulltext) {
```

```
$recipients = "news-alert@example.com";

$headers["From"]    = "news-admin@example.com";
$headers["To"]      = "news-alert@example.com";
$headers["Subject"] = $title;


$params["host"] = "smtp.1and1.com";
$params["port"] = "25";
$params["auth"] = true;
$params["username"] = "user";
$params["password"] = "password";

$mail_text = $title . "\n\n" . $summary . "\n\n" . $fulltext;

$mail_object =& Mail::factory("smtp", $params);
$mail_object->send($recipients, $headers, $mail_text);
}
```

This is another example of how easy it is to add new functionality to PHP programs by reusing existing open source libraries.

Creating PDF files

Lets say that you want to provide PDF versions of certain news to your customers. It is really easy to do so using PHP. In fact there are a great number of PHP libraries that allow you to generate PDF documents dynamically. Some of them require a license for commercial usage. Some others are completely free, such as the fpdf library. This library can be obtained at http://www.fpdf.org/ and is entirely written in PHP, so it is portable and can be used in most. The following code snippet allows you to create a PDF version of a piece of news:

```
require('fpdf.php');

function ($title, $summary, $fulltext) {
      $pdf=new FPDF();
      $pdf->AddPage();
      $pdf->SetFont('Arial','B',16);
      $pdf->Cell(0,10,$title,0,1);
      $pdf->SetFont('Arial','I',14);
      $pdf->MultiCell(0,5,$summary,0,1);
      $pdf->SetFont('Arial','',12);
      $pdf->MultiCell(0,5, $fulltext);
      $pdf->Output();
}
```

Security

So far, we have not considered any security-related aspects of web programming. This has been done for the sake of simplicity. But the moment that you are storing or displaying user-provided data on your web site, you need to take steps towards correctly handling malicious data. It is also often necessary to restrict access to parts of the website to only certain authorized users.

Escaping HTML and SQL

When inserting data into MySQL, there are certain data values that may interfere with SQL syntax, such as the apostrophe ('). This and other characters need to be escaped before being passed to the database. Not only can this break your script in unexpected ways, but a malicious user could exploit these problems to gain access to your data. PHP provides diverse mechanisms to achieve this. Check the PHP manual pages for mysql_escape_string(), addslashes() and get_magic_quotes_gpc() for more information on the topic.

Restricting access

Different access control methods can be implemented at each one of the layers of the LAMP stack. At the operating system layer, using firewall rules. At the web server layer, using Allow/Deny rules and HTTP authentication. At the PHP layer, using form-based authentication. And, finally, at the MySQL layer where users can be granted privileges for only accessing certain databases and tables, for example.

A common example is restricting access to a particular directory or URL. In this case, we want to restrict access to the page that allows news management. We want to restrict access so it is only accessible from a certain range of addresses and only to a particular user that needs to provide a username and password. We can do so with the following Apache configuration snippet :

```
<Location /manage.php>
      Allow from 10.0.0.0/255.255.255.0
      AuthType Basic
      AuthName “News Submission”
      AuthUserFile /usr/local/apache2/conf/htusers
      AuthAuthoritative on
      Require valid-user
      Satisfy all
</Location>
```

You will need to use the Apache `htpasswd` utility to create the appropriate user database htusers.

Conclusion

This tutorial has provided you with an overview of the LAMP platform capabilities, the knowledge of how to set up your LAMP environment, and show you how easy it is to create LAMP based applications. But there is much more to LAMP components such as MySQL and PHP, including clustering, embedding, GUI programming, script compilation. The following URLs are good starting point. Best of luck with your LAMP journey!

MySQL website: http://www.mysql.com
Apache website: http://www.apache.org
PHP website: http://www.php.net
Apache news: http://www.apacheweek.com
Great PHP and MySQL tutorials: http://www.sitepoint.com
Free, easy-to-use LAMPStack distribution: http://www.bitrock.com

Acknowledgements

The material in this tutorial draws from many sources, including the MySQL, PHP and Apache documentation. They are comprehensive, up-to-date references and have played a great part in the success of the platform. In addition to them, you can find more information in other Apache-related books I have authored/coauthored:

Sams Teach Yourself Apache 2 ISBN: 0672323559
Sams Teach Yourself PHP, MySQL and Apache All in One ISBN: 0672327252
Apache Phrasebook ISBN 0672328364 (To be published December 2005)

Contact Info

The author can be reached at daniel@bitrock.com