INSTALLING, CONFIGURING, AND DEVELOPING WITH XAMPP

by Dalibor D. Dvorski, March 2007

Skills Canada – Ontario

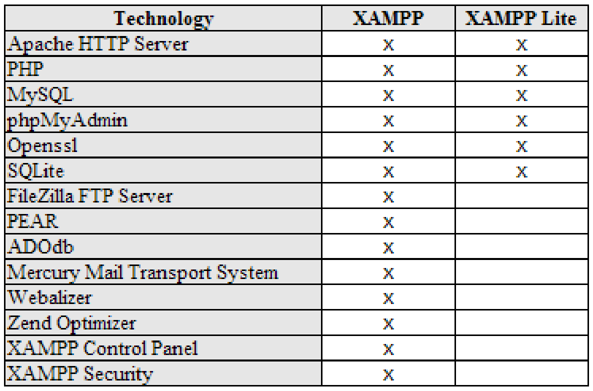
This article explains the installation and configuration process of the XAMPP Apache distribution, and then provides a step-by-step tutorial on developing a simple address book program using the PHP programming language and MySQL database package.

In order to comply with the Ontario Skills Competition, this article will use the following coding standards and tools: XHTML 1.0 Transitional, PHP 5.0, MySQL 5.0, and phpMyAdmin 2.9.

About XAMPP and Installation Requirements

XAMPP is a small and light Apache distribution containing the most common web development technologies in a single package. Its contents, small size, and portability make it the ideal tool for students developing and testing applications in PHP and MySQL. XAMPP is available as a free download in two specific packages: full and lite. While the full package download provides a wide array of development tools, this article will focus on using XAMPP Lite which contains the necessary technologies that meet the Ontario Skills Competition standards. As the name implies, the light version is a small package containing Apache HTTP Server, PHP, MySQL, phpMyAdmin, Openssl, and SQLite. For more details on the packaging and versions, refer to TABLE 1.

In order to be able to successfully run XAMPP Lite, you will require 17 MB for the self-extracting ZIP installation archive and at least 118 MB after it has been extracted on a local hard disk or USB drive.



Technology	XAMPP	XAMPP Lite
Apache HTTP Server	x	x
PHP	x	x
MySQL	x	x
phpMyAdmin	x	x
Openssl	x	x
SQLite	x	x
FileZilla FTP Server	x	
PEAR	x	
ADOdb	x	
Mercury Mail Transport System	x	
Webalizer	x	
Zend Optimizer	x	
XAMPP Control Panel	x	
XAMPP Security	x	

TABLE 1: XAMPP and XAMPP Lite feature comparison chart.

Obtaining and Installing XAMPP

As previously mentioned, XAMPP is a free package available for download and use for various web development tasks. All XAMPP packages and add-ons are distributed through the Apache Friends website at the address: `http://www.apachefriends.org/`. Once on the website, navigate and find the Windows version of XAMPP Lite and download the self-extracting ZIP archive. After downloading the archive, run and extract its contents into the root path of a hard disk or USB drive. For example, the extract path for a local Windows installation would simply be `C:\`.

If extracted properly you will notice a new `xampplite` directory in the root of your installation disk. In order to test that everything has been installed correctly, first start the Apache HTTP Server by navigating to the `xampplite` directory and run the `apache_start.bat` batch file.

NOTE: Depending on your local Windows configuration, you may be asked to provide network access to the Apache HTTP Server. It is important that you allow network access to the server through your firewall in order to have a functioning server.

Next we will test if the server is running correctly by opening an internet browser and typing `http://localhost/` into the address bar. If configured correctly, you will be presented with a screen similar to that of the one in FIGURE 1.

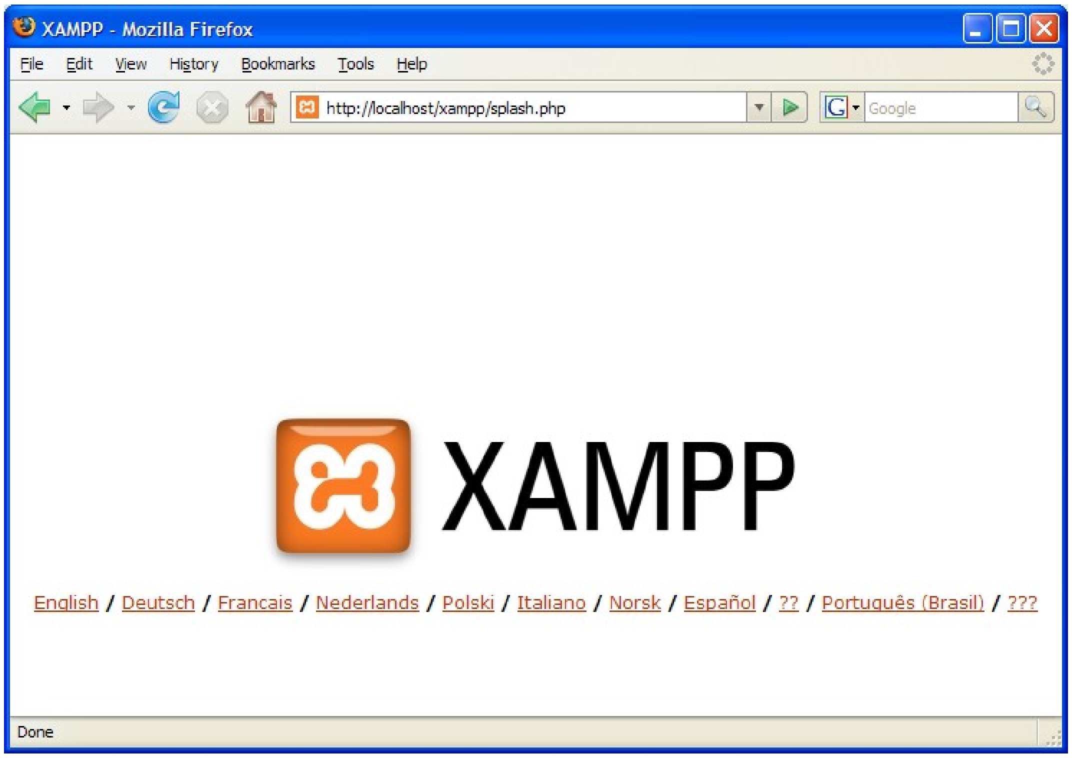

XAMPP - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/xampp/splash.php
Google
XAMPP
English / Deutsch / Francais / Nederlands / Polski / Italiano / Norsk / Español / ?? / Português (Brasil) / ???
Done

FIGURE 1: XAMPP splash screen.

In order to stop all Apache processes do not close the running terminal application, but instead run another batch file in the `xampplite` directory called `apache_stop.bat`.

PHP Hello World

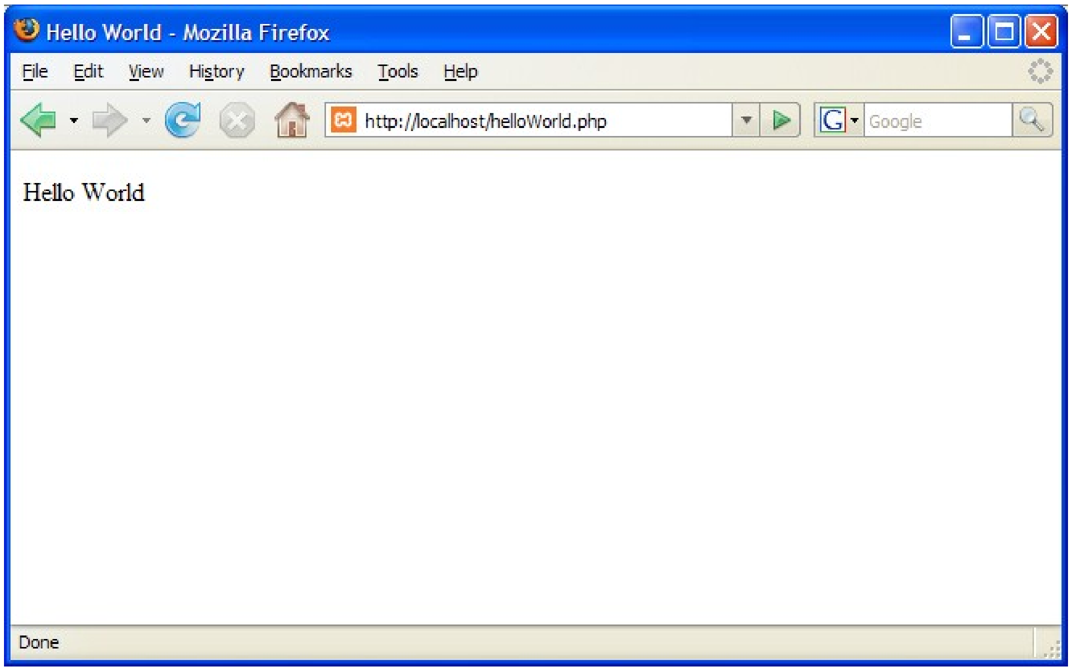
In the previous pages we installed and ran the Apache HTTP Server with success. The next step is to write a simple "Hello World" program in PHP that will test the configuration of PHP under XAMPP.

In a text editor, such as TextPad, write the following lines of code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
    <title>Hello World</title>
</head>

<body>
    <p><?php echo "Hello World"; ?></p>
</body>

</html>
```

The above code example starts a new XHTML document with the page title set as "Hello World" and then prints a single line of text, `Hello World`, using the echo PHP language construct.

In order to run and test this PHP program, save it into your `xampplite\htdocs` directory with the file name of `helloWorld.php`. Then, to view it in your browser, simply type in `http://localhost/helloWorld.php` into the address bar. If done correctly, you will see your `Hello World` line of text in the web browser as shown in FIGURE 2.


Hello World - Mozilla Firefox
File
Edit
View
History
Bookmarks
Tools
Help
http://localhost/helloWorld.php
Google
Done

Hello World

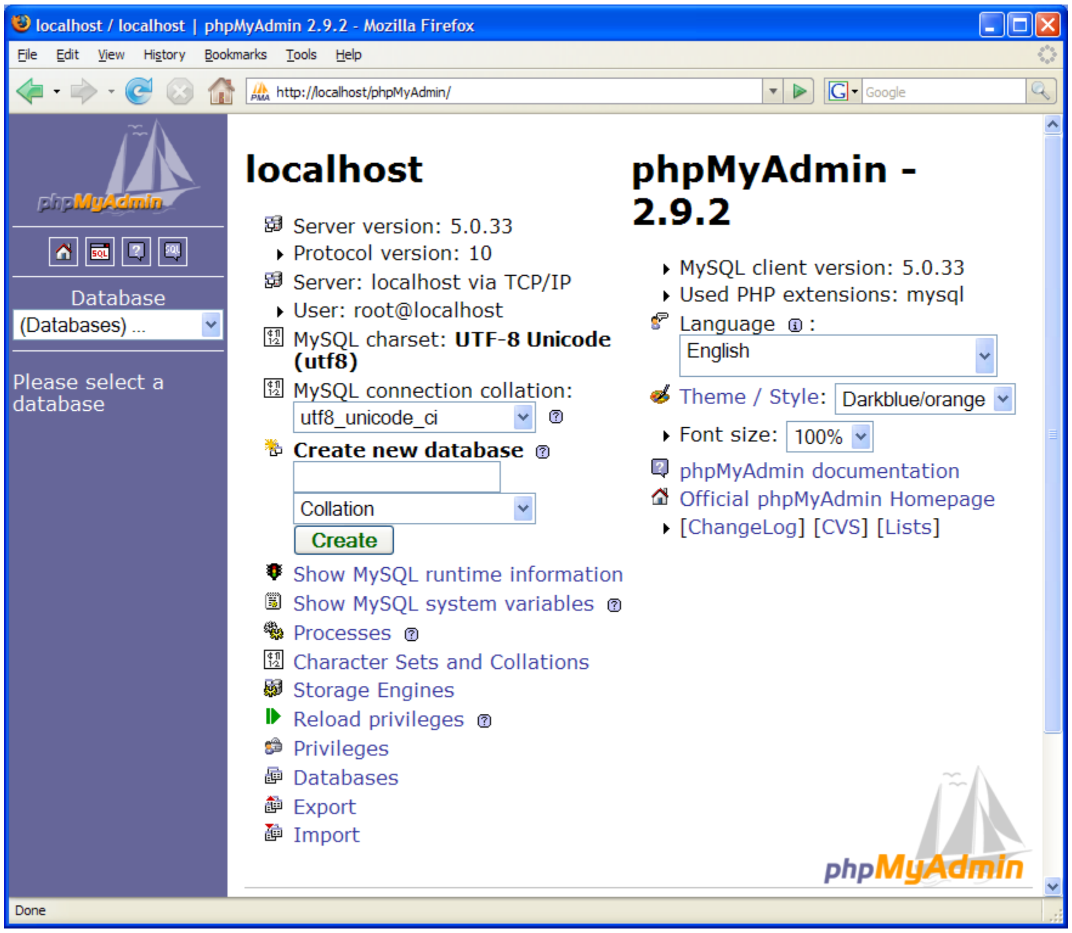
FIGURE 2: Resulting output for helloWorld.php.

Creating a Database and Table, and Inserting Data

Now that we have run and tested Apache and PHP, the next step is running MySQL and creating a database and table which will hold information for our address book program. In order to start MySQL, navigate to the `xampplite` directory and run the `mysql_start.bat` batch file.

NOTE: Similarly to the Apache HTTP Server, MySQL will also need network access; therefore in order to have a functioning database it is important to allow this access through your local Windows firewall configuration.

The XAMPP Lite package contains an application called phpMyAdmin which allows developers to administer and maintain MySQL databases. For the remainder of this article we will be using phpMyAdmin to create a database and table, and enter test data used in the address book program. In order to start phpMyAdmin, you must first copy the `phpMyAdmin` folder from the `xampplite` directory and place it into the `xampplite\htdocs` directory. Before testing phpMyAdmin, make sure that both Apache and MySQL are running by opening their respective batch files: `apache_start.bat` and `mysql_start.bat`. Along with Apache and MySQL running in the background, type `http://localhost/phpMyAdmin/` into your internet browser. If successful you will be presented with a phpMyAdmin start page similar to the one in FIGURE 3.


localhost / localhost | phpMyAdmin 2.9.2 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/phpMyAdmin/
Google
Database
(Databases) ...
Please select a database
Done

localhost

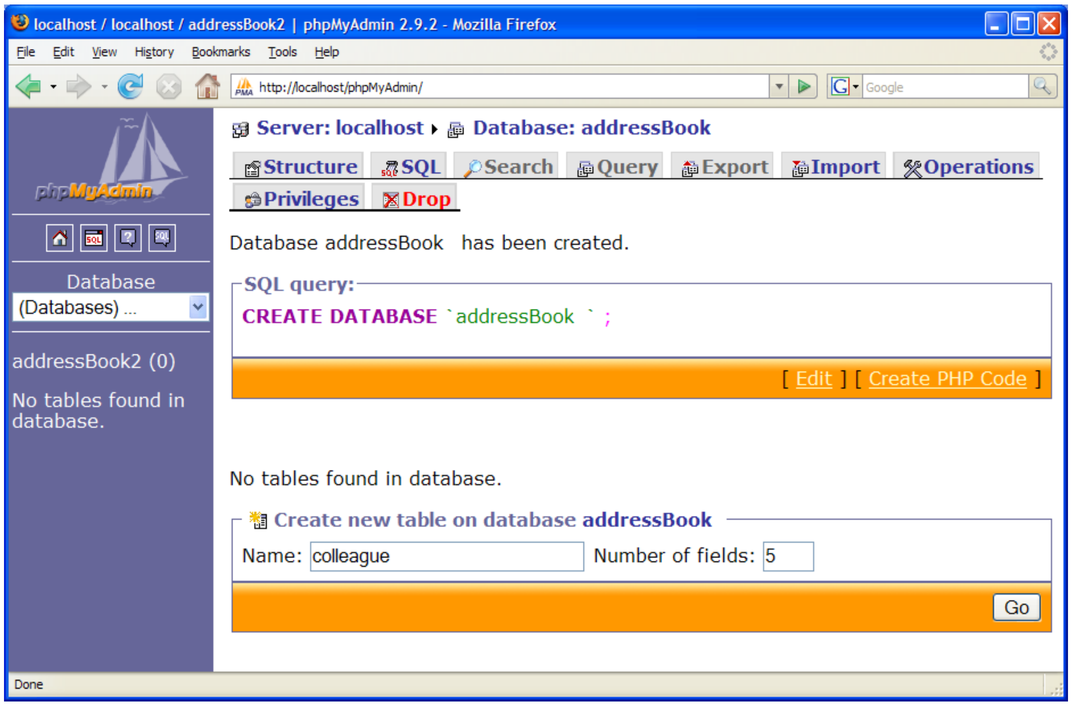
Server version: 5.0.33
Protocol version: 10
Server: localhost via TCP/IP
User: root@localhost
MySQL charset: **UTF-8 Unicode (utf8)**
MySQL connection collation: utf8_unicode_ci
Create new database
Collation
Create
Show MySQL runtime information
Show MySQL system variables
Processes
Character Sets and Collations
Storage Engines
Reload privileges
Privileges
Databases
Export
Import

phpMyAdmin - 2.9.2

MySQL client version: 5.0.33
Used PHP extensions: mysql
Language: English
Theme / Style: Darkblue/orange
Font size: 100%
phpMyAdmin documentation
Official phpMyAdmin Homepage
[ChangeLog] [CVS] [Lists]

FIGURE 3: phpMyAdmin start page.

The first step with phpMyAdmin running is creating a new database. To do this, point to the textbox labelled `Create new database` and type `addressBook` as the name of a new database. Upon doing so, you will be forwarded to a screen similar to the one in FIGURE 4. Next, insert a new table in the `addressBook` database by typing `colleague` into the table name textbox and clicking the `Go` button to proceed with creating the table.


localhost / localhost / addressBook2 | phpMyAdmin 2.9.2 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/phpMyAdmin/
Server: localhost ▸ Database: addressBook
Structure SQL Search Query Export Import Operations
Privileges Drop
Database
(Databases) ...
addressBook2 (0)
No tables found in database.
[Edit] [Create PHP Code]
Go
Done

Database addressBook has been created.

SQL query:

```
CREATE DATABASE `addressBook` ;
```

No tables found in database.

Create new table on database addressBook

Name: colleague Number of fields: 5

FIGURE 4: Creating a new table in the addressBook database.

Following the creation of the `colleague` table, you will be required to insert the columns that make up your table. By using TABLE 2 as a reference, insert the necessary columns that will make up your `colleague` table for the address book.

Field	Type	Length/Values	Extra	Key
id	int	11	auto_increment	primary key
firstName	varchar	20		
lastName	varchar	20		
telephone	varchar	10		
email	varchar	40		

TABLE 2: Table summary for the colleague table.

Lastly, with the database and table created, we next need to input some data into the table. Using the fictional information from TABLE 3 on the next page, input the contact data into your own table.

NOTE: When inputting data into the table, leave the id field blank because it is set to auto increment to an automatic number on every record entry.

id	firstName	lastName	telephone	email
1	James	Smith	1111111111	js@fictionaldomain.com
2	John	Johnson	2222222222	jj@fictionaldomain.com
3	Robert	Williams	3333333333	rw@fictionaldomain.com
4	Richard	Davis	4444444444	rd@fictionaldomain.com
5	David	Brown	5555555555	db@fictionaldomain.com

TABLE 3: Fictional table data for the colleague table.

We are now ready to write the PHP program that will connect to the database and display our data in an internet browser. The procedure that we will write this program is to first establish a connection with the database using PHP, then start our XHTML deceleration and meta data, and finally in the body of our page, write a loop that will iterate through each record in the `colleague` table and display that data in the browser.

```
<?php

    // Connect and select a database
    mysql_connect("localhost", "root", "");
    mysql_select_db("addressBook");

    // Run a query
    $result = mysql_query("SELECT * FROM colleague");

?>
```

The PHP block above will connect to our database using `localhost` as the address, `root` as the username, and a blank password field. This configuration of `root` as the username and a blank password is the default that is set when phpMyAdmin is configured and is what we will be using for our examples.

NOTE: Although it is perfectly fine to use the default username and password for learning purposes, it is strongly suggested to change the password to something more secure if developing full web applications.

Following the `mysql_connect("localhost", "root", "");` line of code is another line stating that we will be connecting to and using the `addressBook` database. Lastly, we will create a variable called `result` which will contain a SQL query that will *select all* records *from* the `colleague` table.

```
<?php

    // Connect and select a database
    mysql_connect("localhost", "root", "");
    mysql_select_db("addressBook");

    // Run a query
    $result = mysql_query("SELECT * FROM colleague");

?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
    <title>Address Book</title>
    <meta http-equiv="content-type" content="text/html;charset=utf-8" />
</head>
```

In the above code example you will notice that the PHP code block, contained within `<?php` and `?>`, is started and then ended before declaring the XHTML document. Remember that in the "Hello World" example earlier in this article we also had a PHP code block within a XHTML paragraph tag. This is important to note because XHTML and PHP can interchangeably be used within one another. This concept will also be shown when we write a PHP loop to retrieve all the records from the database and display it in a XHTML table.

Insert a couple new lines after the closing `</head>` tag and then type in the following code:

```
<body>
    <h1>Address Book</h1>

    <table border="1" cellpadding="2" cellspacing="3"
            summary="Table holds colleague contact information">
        <tr>
            <th>ID</th>
            <th>First Name</th>
            <th>Last Name</th>
            <th>Telephone</th>
            <th>Email Address</th>
        </tr>
```

This new addition to your code will simply start the body of your document and then place a new XHTML table, along with a new table row followed by five table headings: ID, First Name, Last Name, Telephone, and Email Address.

Next we will create a new PHP code block in which we write a `while` loop that goes through each database table record and outputs each piece of data in a new XHTML table cell:

```
<?php

    // Loop through all table rows
    while ($row = mysql_fetch_array($result)) {
        echo "<tr>";
        echo "<td>" . $row['id'] . "</td>";
        echo "<td>" . $row['firstName'] . "</td>";
        echo "<td>" . $row['lastName'] . "</td>";
        echo "<td>" . $row['telephone'] . "</td>";
        echo "<td>" . $row['email'] . "</td>";
        echo "</tr>";
    }

    // Free result memory and close database connection
    mysql_free_result($result);
    mysql_close();
?>
```

This loop will write a new table row for each record by printing the `<tr>` tag and then also write the `id`, `firstName`, `lastName`, `telephone`, and `email` into its own respective table cells using the `<td>` tags.

Finally, before ending the loop, the table row is closed by printing out `</tr>`. The PHP code block is concluded by freeing the memory with `mysql_free_result($result);` and closing the database connection with `mysql_close();`.

The majority of the code has now been written, however, to properly end this XHTML document we will need to insert a few more closing tags:

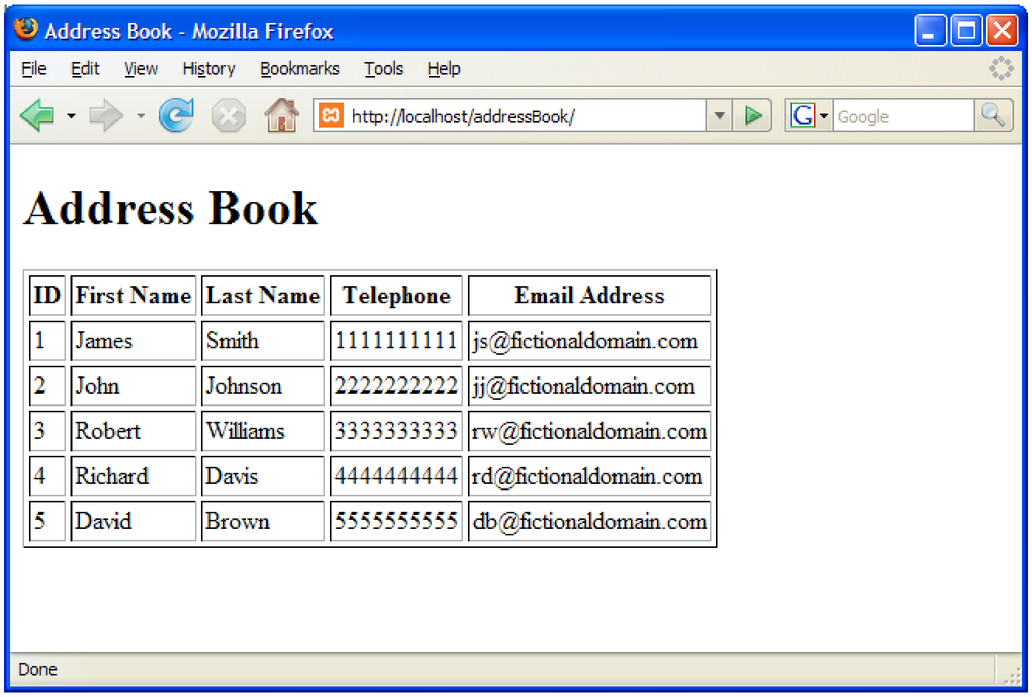
```
    </table>

</body>

</html>
```

You may refer to the next page for complete code of the address book program.

```
<?php

    // Connect and select a database
    mysql_connect("localhost", "root", "");
    mysql_select_db("addressBook");

    // Run a query
    $result = mysql_query("SELECT * FROM colleague");

?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
    <title>Address Book</title>
    <meta http-equiv="content-type" content="text/html;charset=utf-8" />
</head>

<body>
    <h1>Address Book</h1>

    <table border="1" cellpadding="2" cellspacing="3"
            summary="Table holds colleague contact information">
        <tr>
            <th>ID</th>
            <th>First Name</th>
            <th>Last Name</th>
            <th>Telephone</th>
            <th>Email Address</th>
        </tr>

    <?php

        // Loop through all table rows
        while ($row = mysql_fetch_array($result)) {
            echo "<tr>";
            echo "<td>" . $row['id'] . "</td>";
            echo "<td>" . $row['firstName'] . "</td>";
            echo "<td>" . $row['lastName'] . "</td>";
            echo "<td>" . $row['telephone'] . "</td>";
            echo "<td>" . $row['email'] . "</td>";
            echo "</tr>";
        }

        // Free result memory and close database connection
        mysql_free_result($result);
        mysql_close();
    ?>
    </table>
</body>

</html>
```

Now that you have created a database and table, inserted some data into that table, and written a program that reads the data using PHP, all that is left to do is test, and if required debug.

Making sure that the Apache HTTP Server, along with MySQL are running, open up an instance of your internet browser and type in the location of your address book PHP file. For example, if you created an `addressBook` folder inside the `htdocs` folder, then the path to run the address book program would be: `http://localhost/addressBook/index.php`. If done correctly, you will be presented with a page similar to that of the one in FIGURE 5. This figure shows the address book program running with a XHTML table showing the data from the `colleague` table. If you encounter an error, please review and compare your code with the complete code provided on the previous page. Lastly, to stop and close all processes run the `apache_stop.bat` and `mysql_stop.bat` batch files for the server and database respectively.


Address Book - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/addressBook/
Google
Done

Address Book

ID	First Name	Last Name	Telephone	Email Address
1	James	Smith	1111111111	js@fictionaldomain.com
2	John	Johnson	2222222222	jj@fictionaldomain.com
3	Robert	Williams	3333333333	rw@fictionaldomain.com
4	Richard	Davis	4444444444	rd@fictionaldomain.com
5	David	Brown	5555555555	db@fictionaldomain.com

FIGURE 5: The address book program displaying contact information from the database.

Conclusion

Throughout this article you have been shown the beginning concepts that will allow you to set up a development environment, create and administer a database, and write a program that connects to a database and retrieves data.

Although these concepts are important ones, they are in no means the end of an application. By using the address book program as an example, there is still a lot of features that can be implemented in such an application. Features such as editing contact information, deleting contacts, adding new contacts, and being able to email contacts through a form are all additions that can be implemented to the example used in this article. Challenge yourself, use this article and other resources as a jump-start and try adding other capabilities to your own address book program!