EXERCISE OVERVIEW

In order to use PHP and MySQL locally on your computer, you first need to set up a local server. Luckily there are two applications called MAMP and XAMPP that make creating a local server easy. It is very important that you do this exercise, or you will be unable to follow along with the book!

If you are taking this class at Noble Desktop, you can skip this exercise. We've already done the setup for you!

MAC: INSTALLING MAMP/MAMP PRO

For the Mac there are two options for setting up a testing server: MAMP and MAMP Pro. MAMP is the free version of the program and can be used for this entire book—**except for sending and testing email.** MAMP can be used to send emails, but you'd need to manually setup a Postfix server in Apache which is beyond the scope of this book. MAMP Pro makes this easy via a GUI interface and it is the version we use for the class. MAMP Pro also has a few other nice features that make it worth considering if you do a lot of development work and don't want to take the time mucking about in the Terminal.

We'll show you how to install MAMP and MAMP Pro, but keep in mind that if you want to send emails, you'll need MAMP Pro.

1. If you'd like to send emails later in the book install MAMP Pro:

MAMP PRO

1. Launch Chrome.
2. Enter the following URL: **mamp.info/en**
3. Click the **Buy now** button underneath **MAMP PRO: Configure and unlimited number of virtual hosts, DynDNS, E-mail...**
4. Follow the instructions to purchase and download the application.
5. Once the download completes, double-click the downloaded file to unzip it.
6. Open the MAMP_MAMP.PRO pkg file.
7. Follow the instructions for installing MAMP and MAMP Pro. When you're done, the applications will appear in your Applications Folder.
8. Open the **MAMP Pro** application. (Go to Hard Drive > Applications > MAMP Pro and open MAMP Pro.app)
9. The servers should start automatically and a MAMP start page will open up in your web browser. If not, click the **Start** button at the top to start the servers. Then, click the WebStart button to open the MAMP start page.

   NOTE: You can get back to this page by clicking the **WebStart button** in the MAMP application, or by going to **localhost:8888/MAMP** in a browser.

2. If you'd like to use the free version and don't want to send emails or want to set up Postfix manually, use MAMP:

MAMP

1. Launch Chrome.
2. Enter the following URL: **mamp.info/en**
3. Click the **Download now** button underneath **MAMP: One-click-solution for setting up your personal webserver.**
4. Once the download completes, double-click the downloaded file to unzip it.
5. Open the MAMP pkg file.
6. Follow the instructions for installing MAMP. When you're done, the application will appear in your Applications Folder.
7. Open the **MAMP** application. (Go to Hard Drive > Applications > MAMP and open MAMP.app)
8. The MAMP start page will open up in your web browser. If not, click the **Start Servers** button.

   NOTE: You can get back to this page by clicking **Open start page** in the MAMP application, or by going to **localhost:8888/MAMP** in a browser.

WINDOWS: INSTALLING XAMPP

1. Windows users don't need to purchase any upgrade to test email—XAMPP is free and comes with a built-in mail server that makes sending/testing emails easy.

WINDOWS

1. Launch **Chrome.**
2. Enter the following URL: **tinyurl.com/xampp180**

   The download should start automatically (save the file on your desktop if it asks).
3. Once the download completes, double-click the file, then click **Run** to start the install. If it warns about UAC just click **OK.**
4. Click **Next** to start.
5. **C:/** should already be set for the Destination Folder. If not, click Browse and choose it. Click **Next,** then click **Install.**
6. It should go though the install, and when it's done click **Finish.**
7. When it asks to start the Control Panel, click **Yes.** (If it asks to choose a language, choose English.)
8. The XAMPP Control Panel will open.
9. Press **Start** next to **Apache** and **MySQL** to start those services.
10. If you get a warning that Windows Firewall has blocked some features, click **Allow access.**
11. Open Chrome and go to this address: **localhost**
12. Click **English** as your preferred language to finish the installation.
13. XAMPP is now installed and running.

COPYING THE CLASS FILES INTO THE WEB SERVER FOLDER

Throughout this workbook you will be editing class files that we have prepared for you. In order for these PHP files to run, they need to be located in the web server folder that you installed during setup.

1. Go to Desktop > Class Files and follow the appropriate Windows or Mac instructions below:

   MAC: Click once on the **phpclass** folder to select it.
   – Press **Cmd–C** to copy it.
   – Go into **Hard Drive > Applications > MAMP > htdocs**
   – Press **Cmd–V** to paste the class files.

   WINDOWS: Click once on the **phpclass** folder to select it.
   – Press **Ctrl–C** to copy it.
   – Go into **My Computer > C: > xampp > htdocs**
   – Press **Ctrl–V** to paste the class files.

2. Now the class files are in the root directory of our active web server.

MAMP PRO: DISPLAYING ERRORS

By default MAMP Pro does not display PHP errors on the page; instead it just shows a blank document. This is a good secure practice for a production site, but for local development it makes it very hard to troubleshoot your code.

1. Launch **MAMP Pro.**
2. Click the **Server** tab.
3. Click the **PHP** tab.
4. In the **Error Handling** section, toward the bottom under **To,** check on **Display.**
5. Click **Apply.**

EXERCISE OVERVIEW

This exercise gets you started with the basics of PHP syntax. If you're familiar with a language like JavaScript, a lot of this will look familiar to you, but even if you do not have previous programming experience, you'll find that PHP is pretty simple to learn.

MAC: LAUNCH MAMP PRO

1. If MAMP Pro is not already running, go to **Hard Drive > Applications > MAMP Pro** and double-click **MAMP Pro.app**
2. MAMP Pro will launch and it may automatically open the MAMP start page in your default web browser.

   If it does not automatically open, you can click the **WebStart** button in the MAMP Pro application (you may need to click the Start button first), or go to **localhost:8888/MAMP** in a browser.

WINDOWS: LAUNCH XAMPP

If XAMPP is not already running:

1. Click the **Start** button, then choose **Apache Friends > XAMPP > XAMPP Control Panel.**
2. Next to **Apache** click **Start.**
3. Next to **MySql** click **Start.**

ECHO, STRINGS, AND VARIABLES

1. In your text/code editor, open **first.php** from the **phpclass** folder. Remember this folder is located in your htdocs folder here:

   MAC: **Hard Drive > Applications > MAMP > htdocs**
   WINDOWS: **C: > xampp > htdocs**

2. In between the **<body>** tags, add the following bold code:

```
<body>
    <?php
        echo "Hello Universe";
    ?>
</body>
```

   Let's break down this very basic piece of code. We use **<?php** and **?>** to delineate the PHP code from the rest of the page. Everything inside will be PHP, and everything outside will be plain HTML.

   The **echo** command simply prints something to the page. Here we have put a string in double quotes, so it just outputs that string.

The **;** signifies the end of the PHP statement. It is very important not to forget the semicolon or you may get errors in your code.

3. Save the page.
4. Open a browser and go to:
   MAC **localhost:8888/phpclass/first.php**
   WINDOWS **localhost/phpclass/first.php**

   The page should read:

   Hello Universe

5. The echo command will also do simple math. Switch back to your text/code editor.
6. Delete everything between the **`<?php ?>`** tags and add the following code in bold:

```
<?php
   echo 2 + 2;
?>
```

Notice that we did not put quotes around **2 + 2.** This tells PHP to evaluate the numbers as an expression, rather than a string.

7. Save the page and then in a browser go to:
   MAC **localhost:8888/phpclass/first.php**
   WINDOWS **localhost/phpclass/first.php**

   The page should read:

   4

8. Let's see what happens when we add quotes. Switch back to your text/code editor.
9. Add the quotes as shown in bold:

```
<?php
   echo "2 + 2";
?>
```

10. Save the page and then in a browser go to:
    MAC **localhost:8888/phpclass/first.php**
    WINDOWS **localhost/phpclass/first.php**

    The page should read:

    2 + 2

    The quotes tell PHP to evaluate it as a string of text.

11. Now let's make a variable. Switch back to your text/code editor.

12. Delete everything between the `<?php ?>` tags and add the following code in bold:

    ```
    <?php
        $myMessage = "Hello Universe";
        echo $myMessage;
    ?>
    ```

    Don't forget the semicolons at the end of each line!

13. Save the page and then in a browser go to:
    MAC **localhost:8888/phpclass/first.php**
    WINDOWS **localhost/phpclass/first.php**

    The page should read:

    Hello Universe

    You just created a variable called $myMessage and printed it to the page. All PHP variables start with a **$** sign.

    NOTE: Variables are also case-sensitive, so $mymessage and $myMessage would be totally unrelated!

SINGLE QUOTES VS. DOUBLE QUOTES

You may have noticed that we have been using double quotes for our strings. We could also use single quotes, but there are a few subtle differences.

1. Switch back to your text/code editor.
2. Add **single** quotes around $myMessage as shown in bold:

    ```
    $myMessage = "Hello Universe";
    echo '$myMessage';
    ```

3. Save the page and then in a browser go to:
    MAC **localhost:8888/phpclass/first.php**
    WINDOWS **localhost/phpclass/first.php**

    The page should read:

    $myMessage

    The single quotes tell PHP to evaluate everything inside them as a literal string of characters.

4. Now let's try the same thing but with **double** quotes. Switch back to your text/code editor.
5. Add **double** quotes around $myMessage as shown in bold:

    ```
    $myMessage = "Hello Universe";
    echo "$myMessage";
    ```

6. Save the page and then in a browser go to:
   MAC **localhost:8888/phpclass/first.php**
   WINDOWS **localhost/phpclass/first.php**

   The page should read:

   Hello Universe

   Wow, it evaluated the variable. **Double** quotes are a bit more flexible and will allow you to mix variables and strings all at once. Let's try it out.

7. Switch back to your text/code editor.

8. Add the following code shown in bold:

```
    $myMessage = "Hello Universe";
    echo "$myMessage, nice to meet you.";
```

9. Save the page and then in a browser go to:
   MAC **localhost:8888/phpclass/first.php**
   WINDOWS **localhost/phpclass/first.php**

   The page should read:

   Hello Universe, nice to meet you.

ESCAPING CHARACTERS

1. Switch back to your text/code editor.

2. Delete everything between the `<?php ?>` tags and add the following code in bold:

```
<?php
    echo 'It's nice to meet you.';
?>
```

3. Save the page and then in a browser go to:
   MAC **localhost:8888/phpclass/first.php**
   WINDOWS **localhost/phpclass/first.php**

   The page will error out and not work. Why not? PHP sees the single quote in **It's** as the end of the string, and then it gets confused by all the characters after it. We must instead escape the single quote.

4. Switch back to your text/code editor.

5. Add the following \ shown in bold:

```
echo 'It\'s nice to meet you.';
```

6. Save the page and then in a browser go to:
   (MAC) **localhost:8888/phpclass/first.php**
   (WINDOWS) **localhost/phpclass/first.php**

   The page should read:

   It's nice to meet you.

7. We can do the same thing with double quotes. Switch back to your text/code editor.

8. Edit the line so it reads**:**

```
echo "Tell him I said \"Hi\".";
```

9. Save the page and then in a browser go to:
   (MAC) **localhost:8888/phpclass/first.php**
   (WINDOWS) **localhost/phpclass/first.php**

10. It should read:

    Tell him I said "Hi".

    There are other escape characters you can use, such as \\ for a backslash itself, **\t** for a tab, **\n** for a new line, and **\r** for a carriage return.

HEREDOC

Sometimes you want to output quite a lot of text, and adding multiple echo statements and escaping characters would be a real pain. Luckily there is a way for PHP to see an entire block of text as a string that you can even add variables to.

1. Switch back to your text/code editor.

2. Delete everything between the `<?php ?>` tags and add the following code in bold:

```
<?php
echo <<<MYBLOCK
This is my little block of text.<br>
Isn't it nice?<br>
MYBLOCK;
?>
```

The **<<<** tells PHP that a heredoc section is starting. The **MYBLOCK** pieces at the beginning and end act as the delimiter for the block of text. You can choose anything to be a delimiter, but the beginning and ending must match exactly. Also, the ending tag cannot have any spaces or characters in front of it and must be followed by a semicolon.

Notice also that we still need to add <br> tags if we want to make line breaks in HTML.

3. Save the page and then in a browser go to:
MAC **localhost:8888/phpclass/first.php**
WINDOWS **localhost/phpclass/first.php**

It should read:

This is my little block of text.
Isn't it nice?

4. The heredoc behaves like a **double** quoted string, so we can add variables to it as well. Switch back to your text/code editor.

5. Add the following code in **bold:**

```
<?php
$myVar = "You can even put variables in here.";
echo <<<MYBLOCK
This is my little block of text.<br>
Isn't it nice?<br>
$myVar
MYBLOCK;
?>
```

6. Try it out and it should read:

This is my little block of text.
Isn't it nice?
You can even put variables in here.

CONCATENATION

There are a few ways to concatenate text and variables. (Concatenate means to “link things together in a chain or series”.)

1. Switch back to your text/code editor.

2. Delete everything between the **`<?php ?>`** tags and add the following code in bold:

```
<?php
    echo "This ", "is ", "my string.";
?>
```

Pay careful attention to the spaces when you type this!

3. Save the page and then in a browser go to:
MAC **localhost:8888/phpclass/first.php**
WINDOWS **localhost/phpclass/first.php**

It should read:

This is my string.

4. You can do the same thing with a **period.** Switch back to your text/code editor.

5. Change the commas to **periods** as shown in **bold:**

```
echo "This ". "is ". "my string.";
```

6. Try it out and it should read:

This is my string.

7. The period syntax is a little bit more flexible. With it we can append something directly to a variable. Switch back to your text/code editor.

8. Delete everything between the **`<?php ?>`** tags and add the following code in bold:

```
<?php
    $msg = "I like carrots.";
    $msg .= " And broccoli.";
    echo $msg;
?>
```

9. Save the page and then in a browser go to:
MAC **localhost:8888/phpclass/first.php**
WINDOWS **localhost/phpclass/first.php**

It should read:

I like carrots. And broccoli.

The **.** appends the string to the variable. This is a useful shorthand and you will see it often.

COMMENTS

1. Switch back to your text/code editor.
2. To comment out a single line of code, add **//** to the beginning of the line. Add the following code in bold:

   ```
   $msg = "I like carrots.";
   //$msg .= " And broccoli.";
   echo $msg;
   ```

   In most text editors, the middle line will gray out, indicating that it is a comment and will no longer be executed.
3. To comment out multiple lines use **/*** and ***/** as shown in **bold** below:

   ```
   /*$msg = "I like carrots.";
   //$msg .= " And broccoli.";
   echo $msg;*/
   ```

   Now the entire block of code will gray out indicating it is one big comment.
4. Close the file, saving changes if you wish. We're done with it.