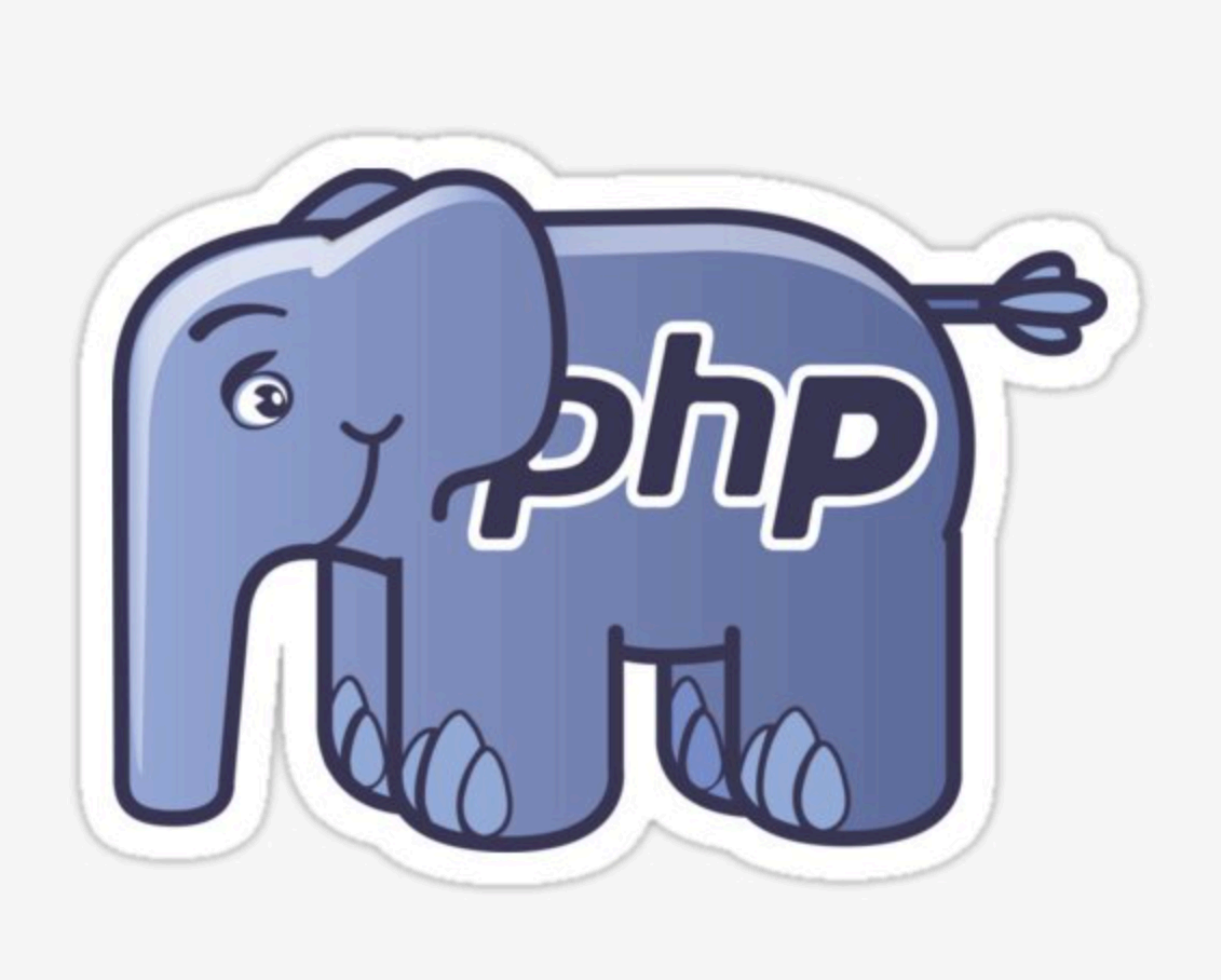
Week 1 - Intro to PHP

Course outline & review

This course will deliver learning opportunities to practice new skills through hand-on projects based on the content delivered in HTTP 5211 Web Development 1. This lab experience will allow students to simulate the daily activities of a Web Developer and will build on the programming languages, database development and toolkit introduced in semester one and required by an entry-level web developer and include markup language, web services, data exchange applications, server-side technology to support the generation of dynamic web-based applications and more.

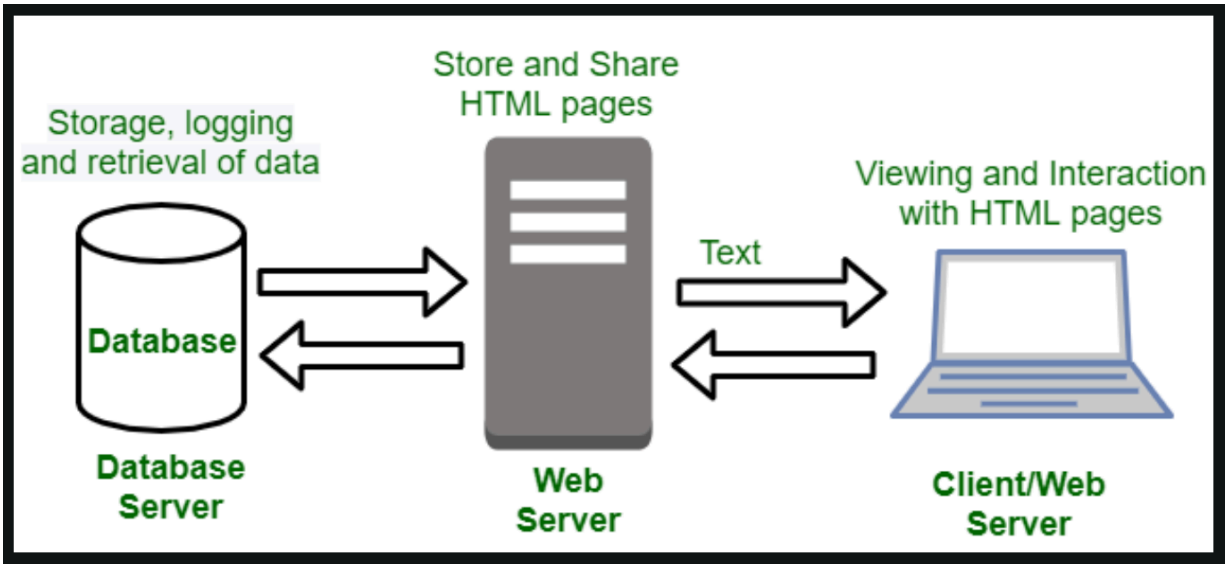
Review of PHP



PHP is a scripting language. PHP is an acronym that stands for Hypertext Preprocessor. PHP is a widely used programming language for web development. Particularly, we can use PHP to make webpages dynamic. It was created in 1994 by Ramus Lerdorf as a set of Common Gateway Interface (CGI) scripts to track visitors to his website. Since then it has evolved into a full fledged programming language.

In the current day, 80% of websites on the entire internet rely on PHP to function!

One of the key strengths of PHP is that we can use it to interact with databases. As stated earlier, PHP can be used to generate dynamic data and this is how we can do that. By Creating Reading Updating and Deleting (CRUD) data on a database.



PHP is a server side scripting language. What does that mean? To understand this, let's start from the top. When you view a website: Your computer sends a request to the server,

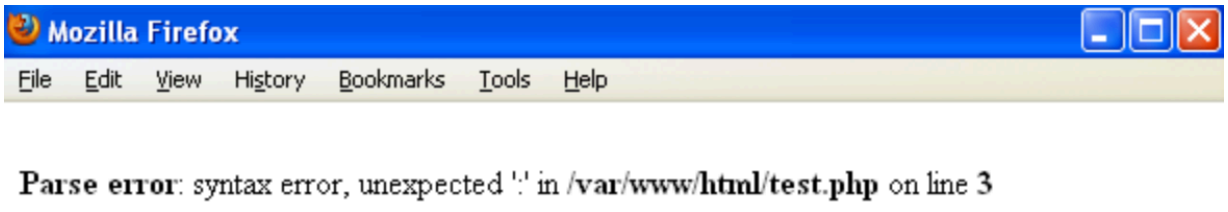
requesting the HTML and any other associated scripts. The server grants tis request and sends the HTML, CSS and the JS associated with the webpage. These scripts are interpreted on **your** computer. In other words it is downloaded and opened on your end. In the case of a static website this also includes the content.

However, with PHP. The code is never sent to the clients computer. It is executed on the server. This has positive implications. For example: If you needed to send form data that contained sensitive data it would be harder to intercept because the code that is handling that data is stored on the server and is never sent to the users computer. It is behind a password protected gateway on the server. Great for security.

Various content management systems also run on PHP. Wordpress, Joomla, PHPBB, E-commerce shops, you name it! There are plenty open source solutions written in PHP that you can use to meet the needs of any modern website from personal small sized sites, all the

way up to enterprise level.

PHP is not a compiled programming language like C but it does have to be correct for it to work. If there is errors in your PHP code, your page will throw an error.



When you throw an error code it will tell you where the error is:

By file name: (/var/www/html/test.php)

And line of code: (on line 3)

Server Side Development Environment (FTP, MySQL, PHPMyAdmin)

As we know by now. When it comes to HTML, CSS and JS, we can just start writing it with an IDE and run it in any browser and it will work.

We can do this If the file is saved on our local computers or upload it to the server and it will work either way.

PHP is a little different. We can't just write PHP and expect it to work. As stated earlier, PHP is server side script. We need to run it on a server, or a server environment running on our computer.

Local Environment

To run PHP on out computers we can use XAMPP.

https://www.apachefriends.org/

What is XAMPP?

XAMPP is the most popular PHP development environment

XAMPP is a completely free, easy to install Apache distribution containing MariaDB, PHP, and Perl. The XAMPP open source package has been set up to be incredibly easy to install and to use.

What this allows us to do is a build of an apache environment on our computers locally. This will give us a space to work and run PHP and SQL databases. Now remember this is running *locally* so what we run on here will not be accessible to the web!

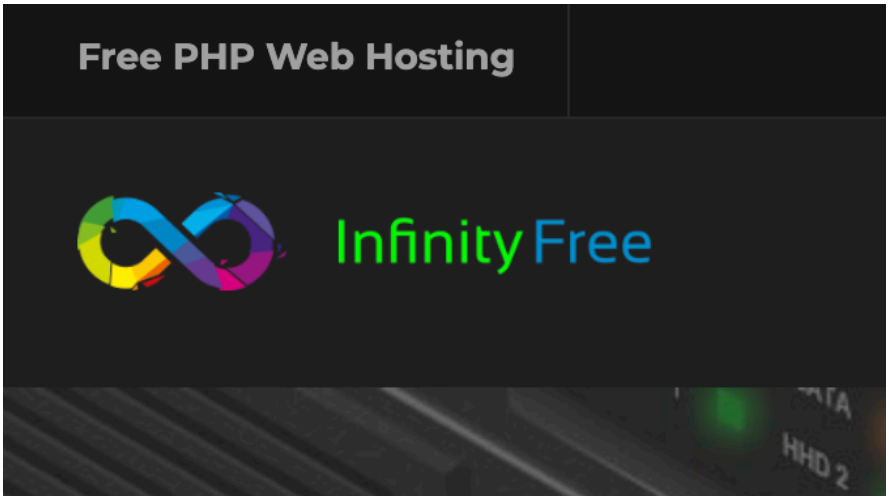
Web Environment

To run PHP that will be accessible to the web (like on a launched website) we have to use web hosting.

There are some solutions we can utilize for free:

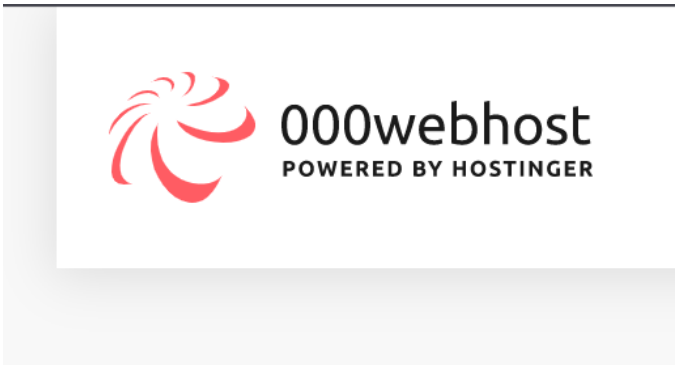
Infinity Free

https://www.infinityfree.net/



000 Web Host

https://www.000webhost.com/



Are just a couple to get us started. Here we can

run a web hosting environment that runs PHP and MySQL (database).

Sample PHP File

Another thing you are used to, is saving an HTML file as a .html this will not work if you want to run PHP.

somephp.html

somephp.html > html > body > ?

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Document</title>
8   </head>
9   <body>
10      <?php
11
12      echo '<h1>Some Text!</h1>';
13
14      ?>
15  </body>
16  </html>
```

In the above example, the file somephp.html will not run the php script. It will see the <?php tag and try to render it like a regular html tag.

Some Text!'; ?>

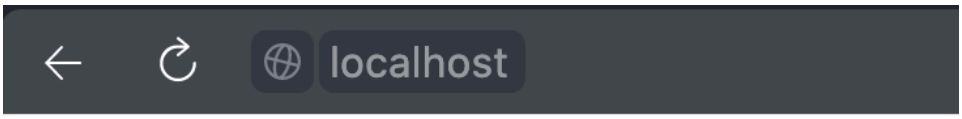
We need to save the file as a PHP file for it to work:

somephp.php

somephp.php > html

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Document</title>
8   </head>
9   <body>
10      <?php
11
12      echo '<h1>Some Text!</h1>';
13
14      ?>
15  </body>
16  </html>
```

Now it will render, so long as we run this in XAMPP or on our server.



Some Text!

As you can see above, the .php file will also render all the HTML, we write it just like an HTML file but with the added functionality of being able to run PHP in the HTML. So, any HTML file we will need to run PHP in, we will simply need to use the .php extension instead of the .html extension, from there we can build an HTML page as we usually would but also be able to add PHP in it. We can also write .php files that only contain PHP in them.

Let's take a quick and simple look under the hood.

First off. To write PHP you have to use a PHP tag. <?php ?> like this.

All PHP code should be within these tags.

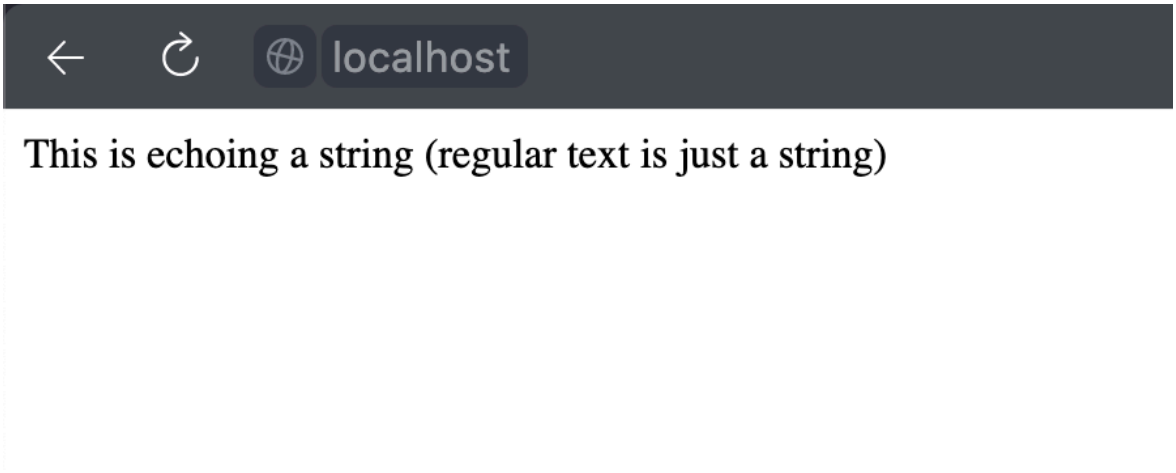
We can then write something very simple, like:

```
<?php

    echo “The thing you want to echo”;

?>
```

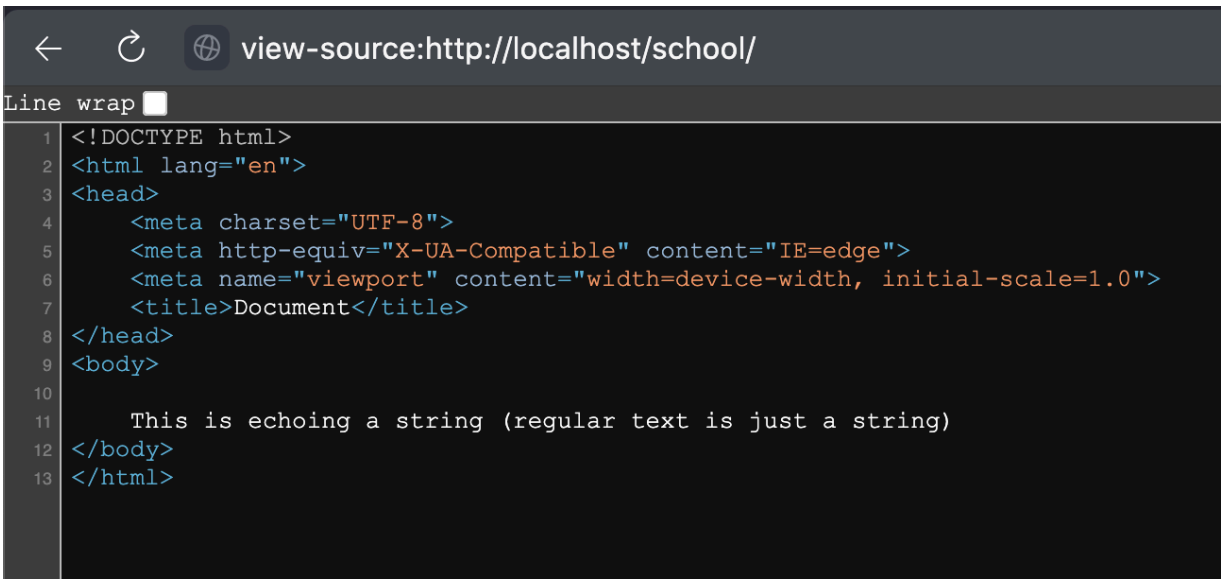
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>

    <?php
        echo "This is echoing a string (regular text is just a
string)";
    ?>

</body>
</html>
```

Now look and see what it looks like in the

browser:



There is the string. If we look at the source code, we will see that there is no PHP code, The browser will not show the PHP code only the result. Which again is great for security and privacy.



In the case above the text renders as plain text. We can add HTML in what we want PHP to return and the browser will render the HTML:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>

    <?php
        echo "<h1>This is echoing a string (regular text is just
a string)</h1>";
    ?>

</body>
</html>
```

Now look again:

It has now rendered the H1 tags we added and passed through the PHP.

Line wrap

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

    <h1>This is echoing a string (regular text is just a string)</h1>
</body>
</html>
```

Looking into the source code we can see that they are rendered.

We can do some more stuff:

```
<body>

    <?php
        echo "<h1>This is echoing a string (regular text is just
a string)</h1>";
    ?>

    <?php
        echo 9*3
    ?>
```

We can run some math, in this case which will return:

We can make variables and echo them:

```
<!-- Below we create a variable -->
<?php
    $name = "Anthony";
?>

<!--//and below we echo the result of the variable-->
<h1>Hello <?php echo $name; ?>. How are you?</h1>
```

The name below is the result of echoing out a variable

Hello Anthony. How are you?

These are just a couple examples. When it comes to programming there are different data types. We will cover this more in depth in later

classes.

Data Type	Represents	Examples
integer	whole numbers	`-5`, `0`, `123`
floating point (real)	fractional numbers	`-87.5`, `0.0`, `3.14159`
string	A sequence of characters	`"Hello world!"`
Boolean	logical true or false	`true`, `false`
nothing	no data	`null`

More on XAMPP

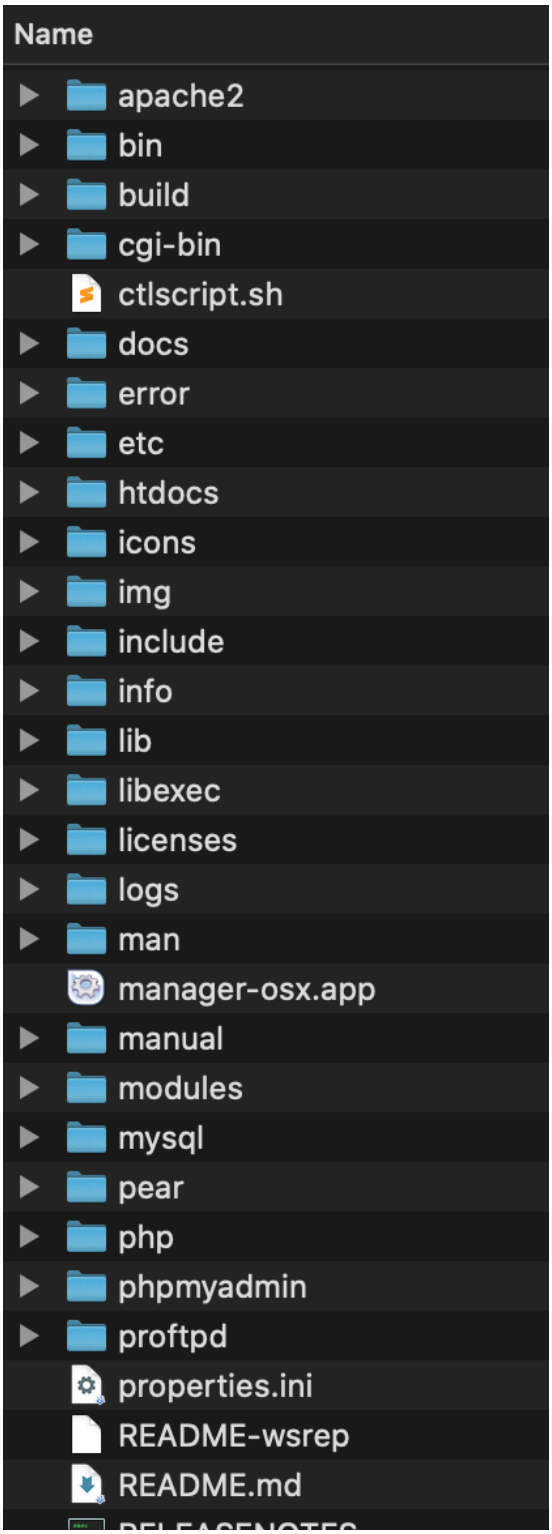
By now you have downloaded XAMPP. Let's take a quick look around.

When you launch the application you will see this:

From this screen we can do a few things. First off, if you click “Go to Application” it will open a web browser and load the index.html in the HTDOCS folder within your server environment.

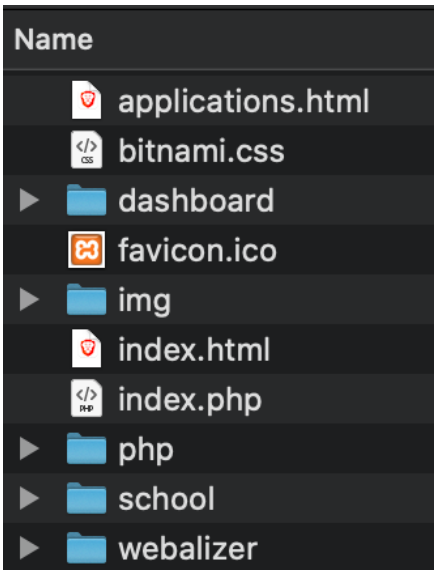
Which brings us to the next part we need to understand. Like stated earlier XAMPP is like running your own apache server on your computer. As such there is a file system within it. You should be saving your files within it. Click the “Open Application Folder” button.

It will open this:



Within this file system, you want to be working in the “htdocs” folder for the most part. This is no different than when working on an actual hosting server. The folder to publish web pages will either be named public_html or htdocs.

In the htdocs folder you will see this (minus the “school” folder):



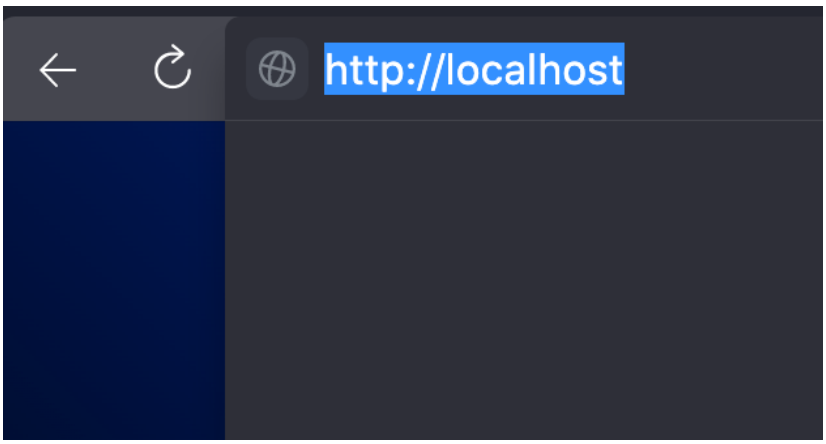
If you were to change the content of the

“index.html” folder it will change what you see when you click the “Go to Application” button.

In here we can add more folders and files, the same as we would when working on an actual apache server.

Im my case I have added a “school” folder and inside that folder it has all the different courses. Remember to stay organized!

When navigating to pages within the server, you have to go to “localhost” to do so.

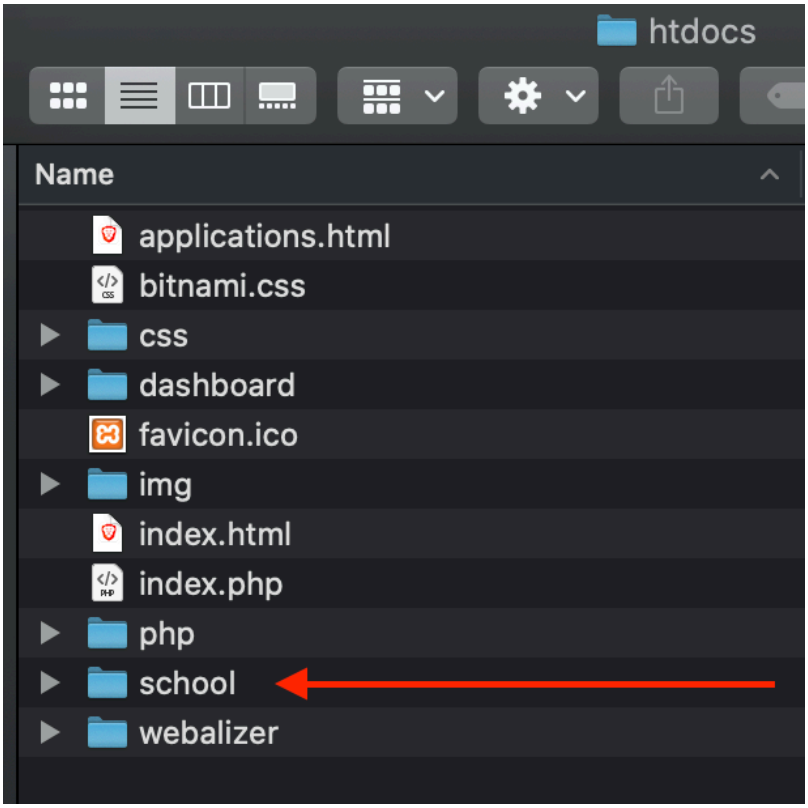


This is your home page. Which resolves to: http://localhost/index.html

If you want to make a child folder, you can do

so inside the “htdocs” folder. For example:

Make a folder like mine, names “school”.

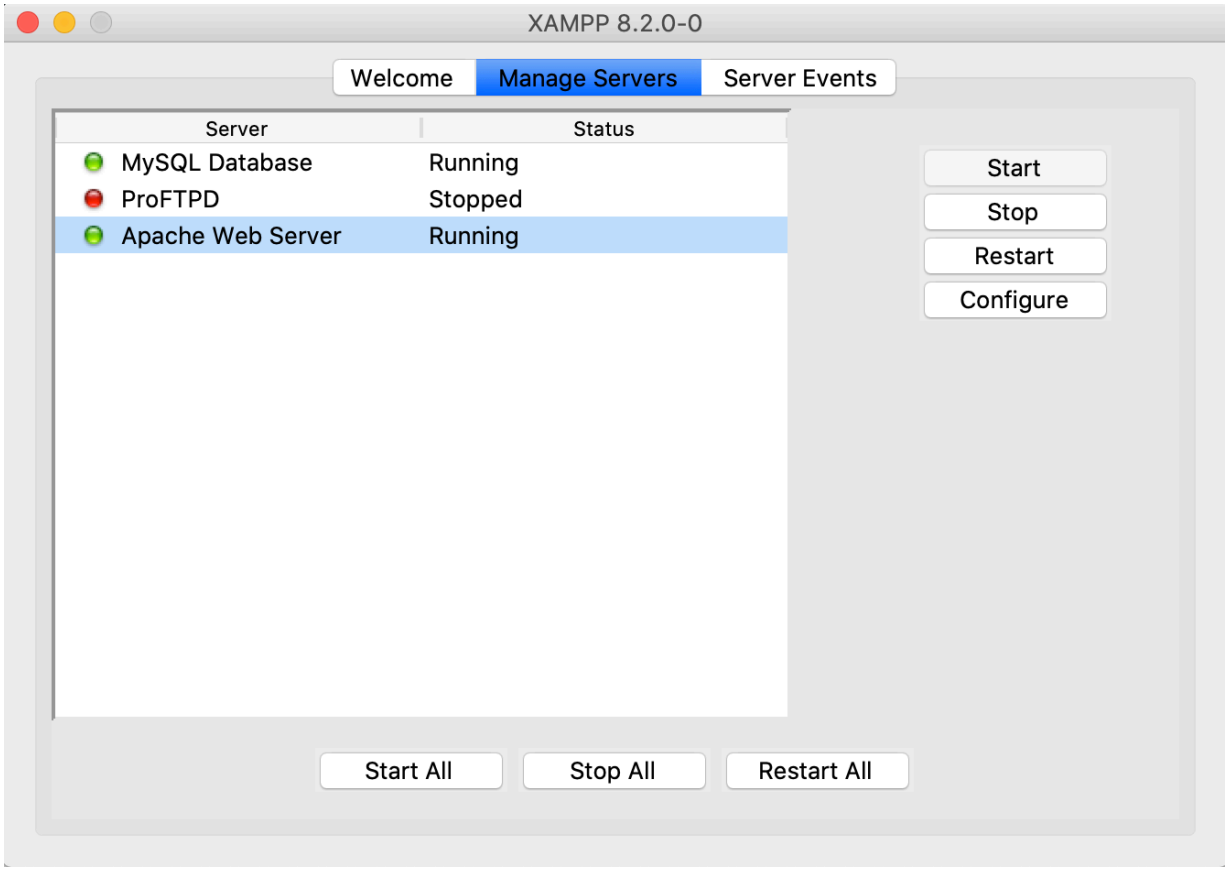


If you were to make a file named index.php in the “school” folder, then the URL would be:

http://localhost/school/

Which would resolve to:

http://localhost/school/index.php



If you click “Manage Servers” you will see this screen. Here you can, as the name states manage servers. The Apache Web Server must be running for you to be able to run the files saved in it.

The MySQL Database must be running to access the DB’s.

You can connect to the Apache server via FTP,

if you wish to you must enable it. Be advised this can allow incoming connections.

Natural evolution of website technology

Why use all this? As you know by now, we *can* make static websites but they can be hard to maintain.

For one, we would need to manage all the HTML files and content. So if we needed to add a new page like on a blog, we would need to go copy the template page, write our content and then save it and link it throughout the website. This would be a real pain.

Another problem among many is we would have to also write HTML constantly, not everyone knows how to do this. Imagine owning a business, not knowing HTML and having to update the site.

With PHP we could instead just load the same 1 HTML page and change the content within the 1

page. Furthermore we could run a more robust solution like Wordpress or Joomla (A CMS System) which we could install, configure and update when needed.

This would be the natural evolution of web applications.

The solutions we have today allow anyone to run and maintain websites, from e-commerce to blogs, personal websites all the way up to enterprise and fortune level corporate concerns. Once the site/web application is created and deployed. Anyone with minor technical knowledge can update it.

This is not to take away from the DEV's though because they are still needed on the back end to make sure everything is running.