COMP4021
Internet Computing

Basic PHP

Gibson Lam and David Rossiter

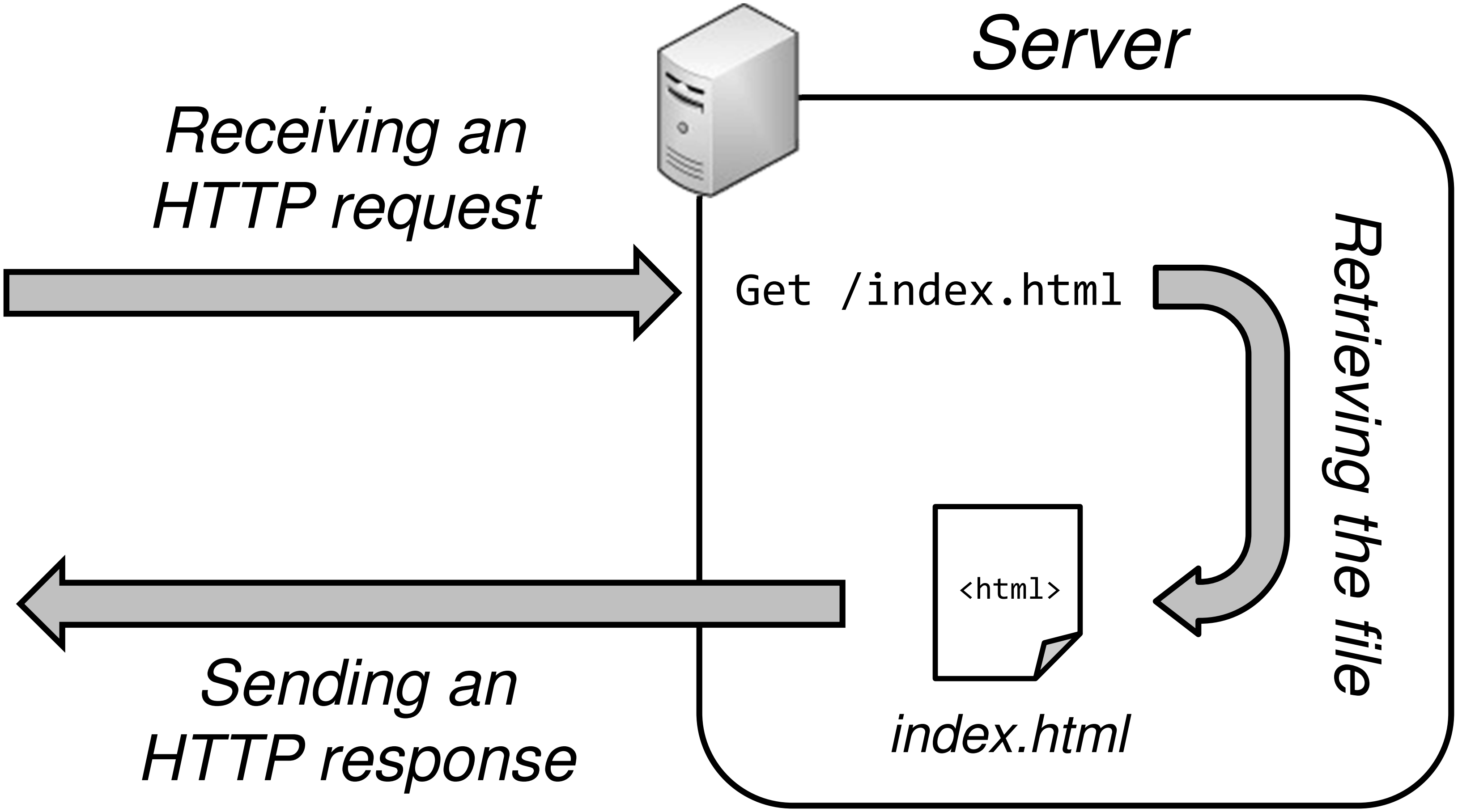
PHP

- PHP stands for
  PHP: **H**ypertext **P**reprocessor
- It is a very popular scripting language used in web servers
- PHP is used by 78% of all websites which use a server-side programming language (April 2022) *
- PHP is not hard to learn

* From https://w3techs.com/technologies/details/pl-php

Requesting a Static File

- Here is an example of what happens when the browser requests for a static file:

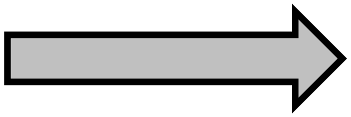


An Example HTML File

- On the left is an example HTML file stored in the server; when the browser requests for and receives the HTML file, the file is the same as the one on the server

```
<!DOCTYPE html>
<html>
<head>
    <title>My Page</title>
</head>
<body>
    <p>Hello!</p>
</body>
</html>
```

HTML file on the server



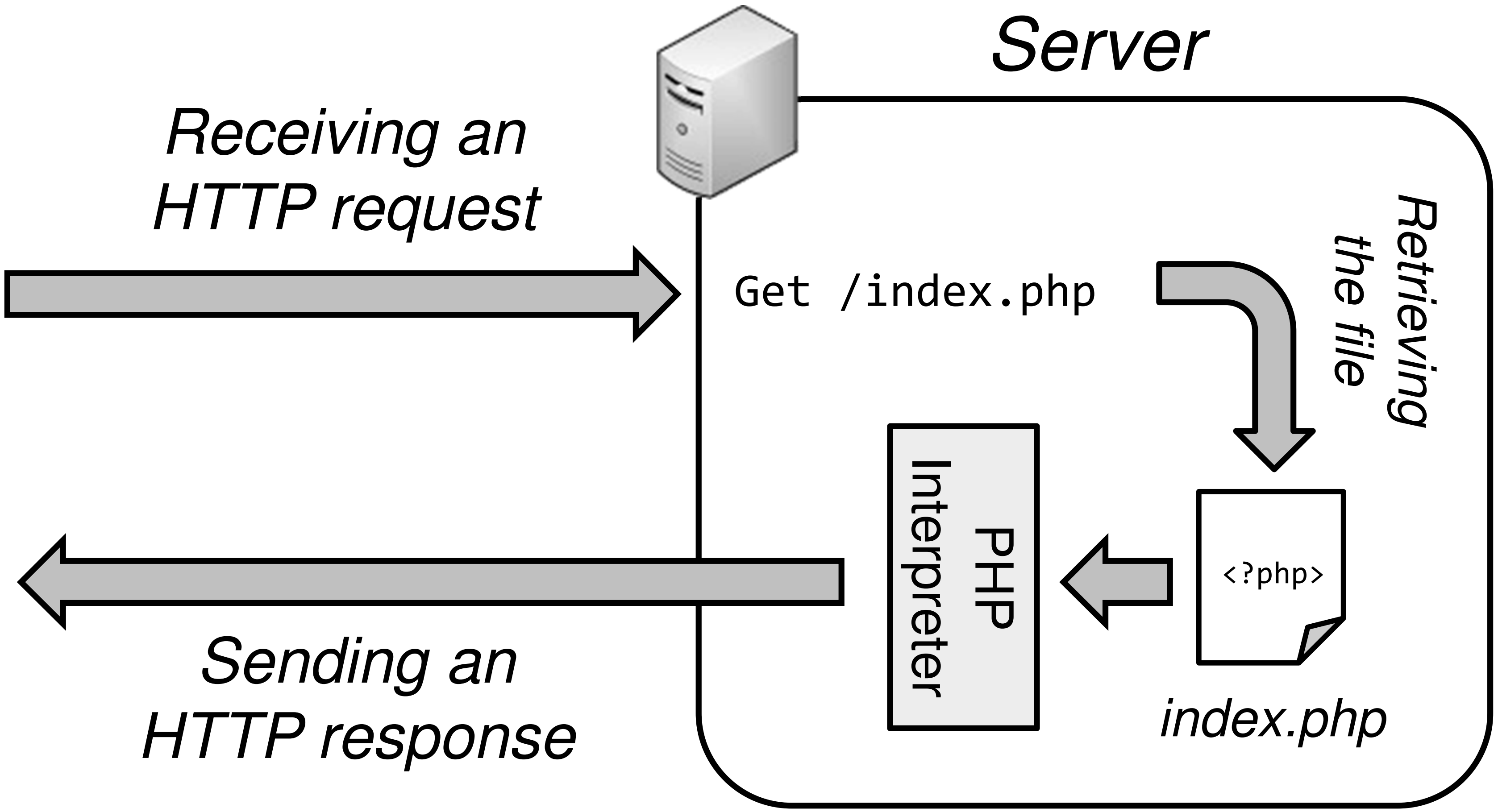
Sending the file from the server to the browser

```
<!DOCTYPE html>
<html>
<head>
    <title>My Page</title>
</head>
<body>
    <p>Hello!</p>
</body>
</html>
```

HTML file received by the browser

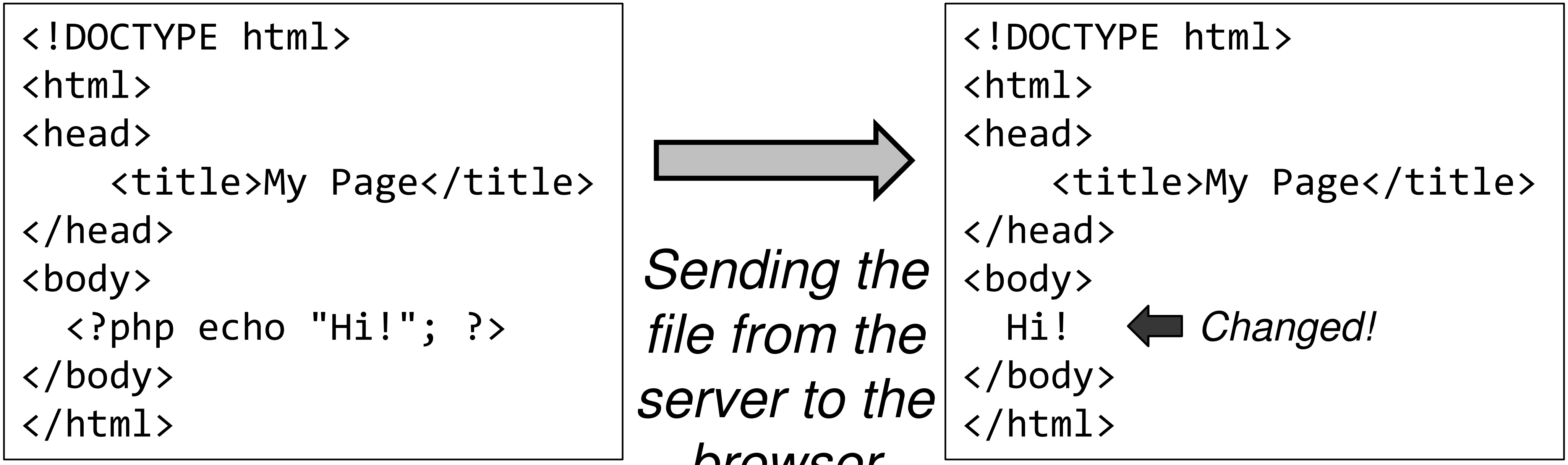
Requesting a PHP File

- Here is an example of what happens when the browser requests for a PHP file:



An Example PHP File

- On the left is an example PHP file stored in the server; when the browser requests for and receives the PHP file, the file is changed by PHP before it arrives



```
<!DOCTYPE html>
<html>
<head>
    <title>My Page</title>
</head>
<body>
  <?php echo "Hi!"; ?>
</body>
</html>
```

PHP file on the server

Sending the file from the server to the browser

```
<!DOCTYPE html>
<html>
<head>
    <title>My Page</title>
</head>
<body>
  Hi!
</body>
</html>
```

Changed!

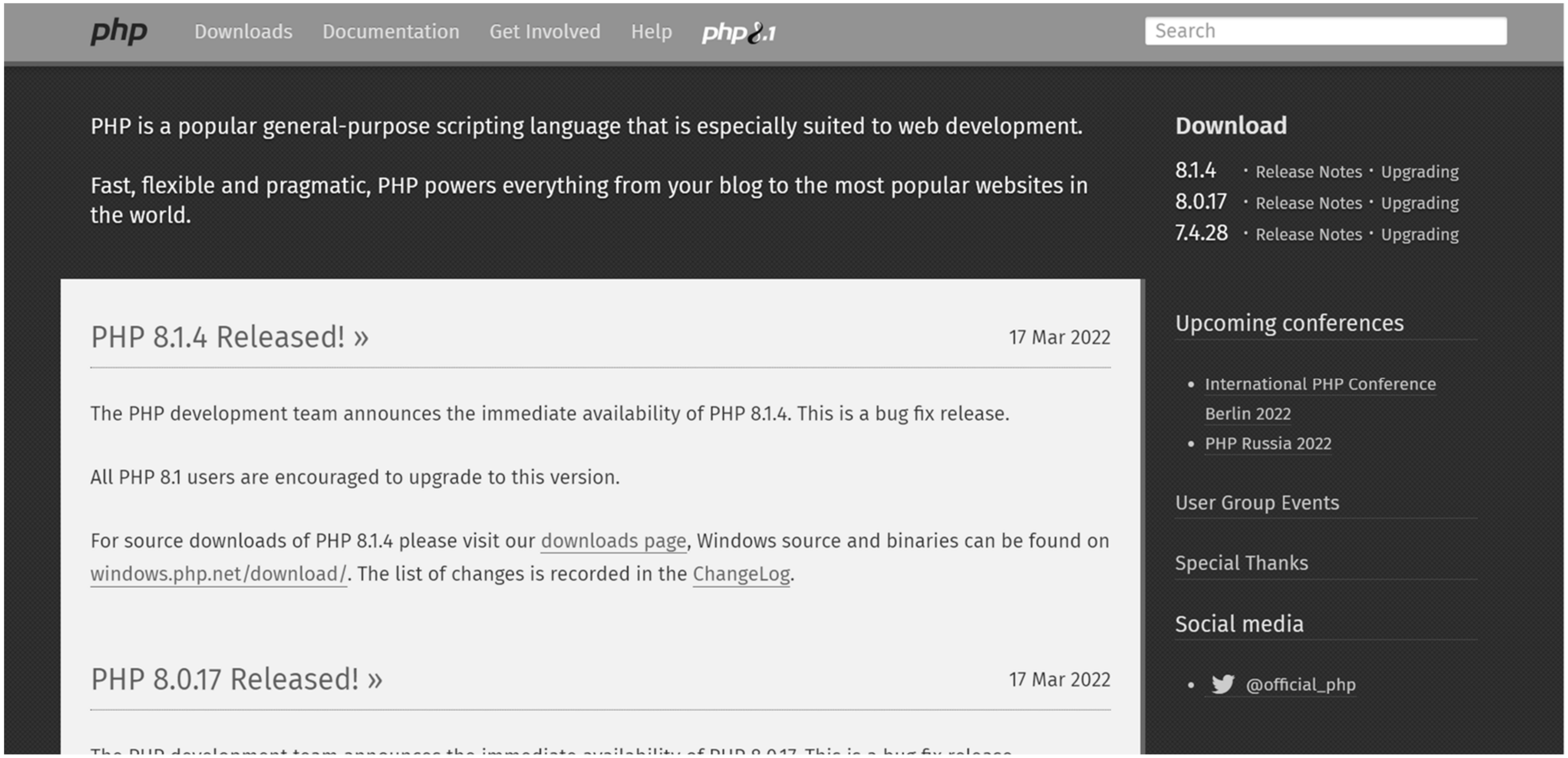
HTML file received by the browser

Downloading PHP

- You can download PHP from its official website:

  `https://php.net`

- The latest version is PHP 8



PHP is a popular general-purpose scripting language that is especially suited to web development.

Fast, flexible and pragmatic, PHP powers everything from your blog to the most popular websites in the world.

PHP 8.1.4 Released! »

17 Mar 2022

The PHP development team announces the immediate availability of PHP 8.1.4. This is a bug fix release.

All PHP 8.1 users are encouraged to upgrade to this version.

For source downloads of PHP 8.1.4 please visit our downloads page, Windows source and binaries can be found on windows.php.net/download/. The list of changes is recorded in the ChangeLog.

PHP 8.0.17 Released! »

17 Mar 2022

The PHP development team announces the immediate availability of PHP 8.0.17. This is a bug fix release.

Download

8.1.4 · Release Notes · Upgrading
8.0.17 · Release Notes · Upgrading
7.4.28 · Release Notes · Upgrading

Upcoming conferences

- International PHP Conference Berlin 2022
- PHP Russia 2022

User Group Events

Special Thanks

Social media

- @official_php

Setting Up PHP

- After extracting the PHP package, you then need to set up your web server to use PHP
- The configuration is different for different web servers
- If you don’t want to do much configuration, you can:
  - Use a ready-made package, or
  - Use the PHP built-in web server

Ready-made Packages

- If you don’t have any web servers installed, you can use special packages to get lots of things installed at the same time
- One popular package is XAMPP, which includes some main server components:
  - An Apache web server
  - A database
  - PHP
  - Sometimes other things as well

XAMPP

Using the PHP Web Server

- PHP provides a built-in web server for quick testing of PHP code
- You can start the server using the following command:

```
php -S localhost:8000
```

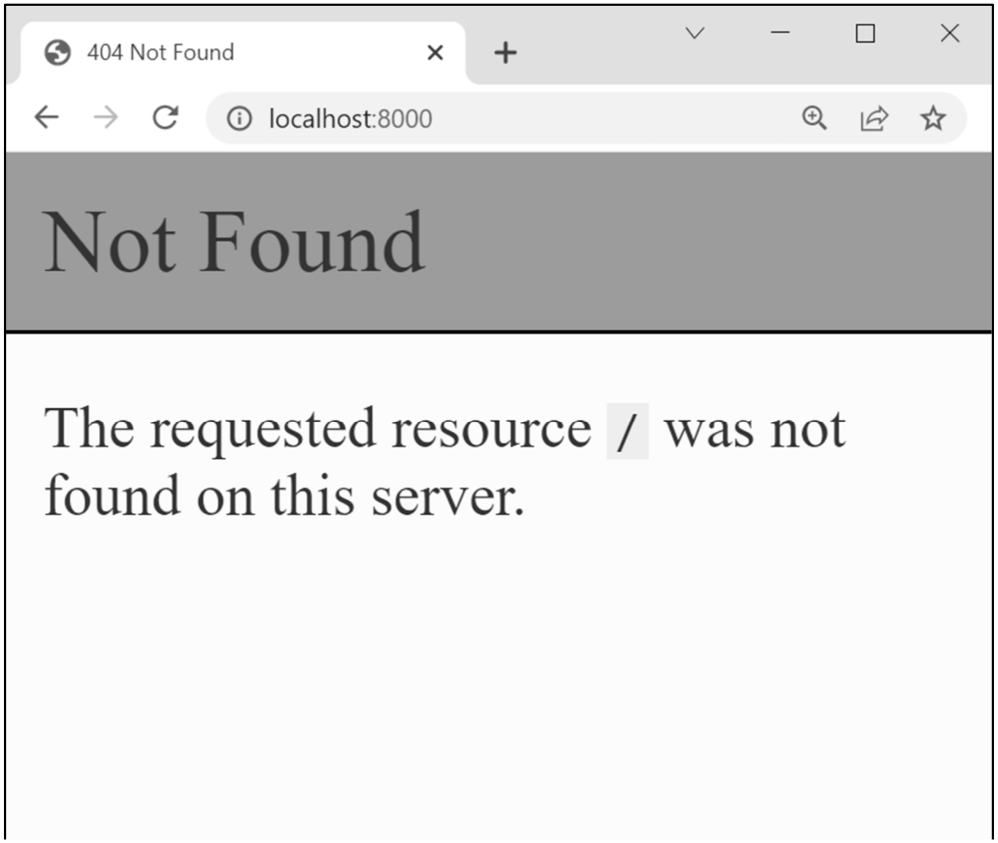
- It runs a web server at `localhost`, i.e. your local computer, using port 8000
- You can then access the server using this URL: `http://localhost:8000`

Browsing the Server

- Let’s start the server in the folder containing the PHP files, e.g. `C:\php` :

```
C:\php>php –S localhost:8000
```

- Although the server is running, you will see “Not Found” shown on the page:
- This is because the folder does not have any PHP files



Not Found

The requested resource `/` was not found on this server.

The Document Root

- You need to start the PHP server in the folder containing your PHP files, or use the `-t` option

```
C:\php>php –S localhost:8000 –t C:\MyWebSite
```

Document root

- The above command runs the PHP server using `C:\MyWebSite` as the document root, i.e. the top level folder of your website

A Simple PHP Example

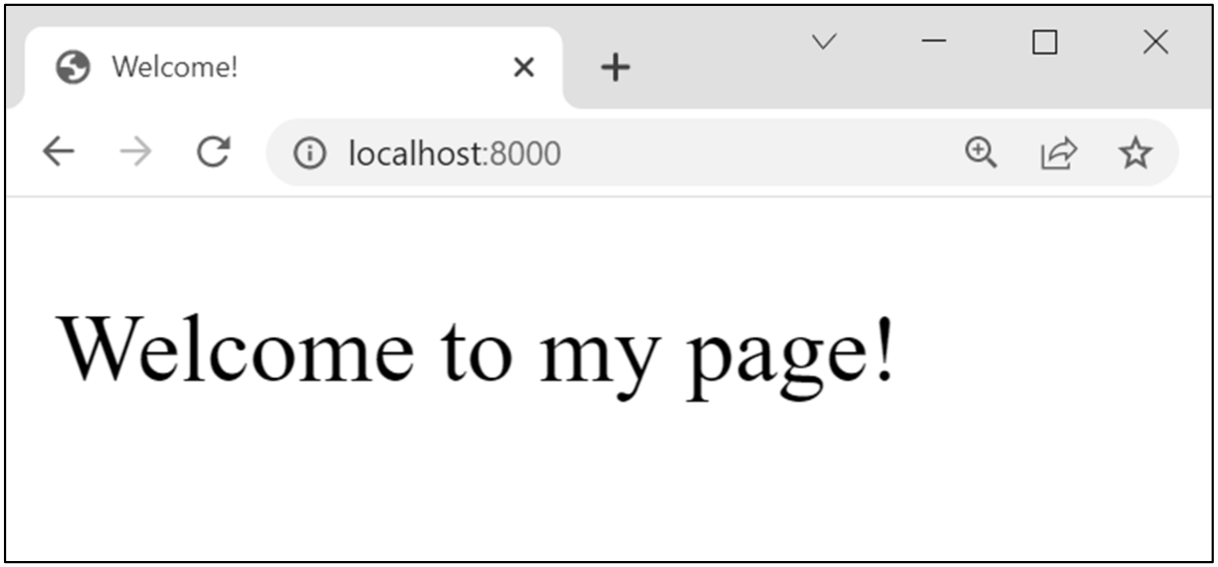
- Let’s put this example PHP file with the file name `index.php` in the document root of your website:

index.php

```
<!DOCTYPE html>
<html>
<head>
  <title>Welcome!</title>
</head>
<body>
  <?php echo "Welcome to my page!"; ?>
</body>
</html>
```

Showing the Example

- Using the same URL, you will see this result:



Welcome to my page!

- You do not need to enter the file name in the URL because `index.php` is one of the default files used when you request for the root path, i.e. /
- You will need to specify the file name when you refer to other PHP files

PHP Scripts

- PHP scripts are enclosed within a special tag `<?php   ...   ?>`
- You can write as many lines of code as you want inside a tag
- When the PHP file is sent to the PHP interpreter, the interpreter runs the PHP code inside the tags and replaces the tags with the output of the code

Typical Use of PHP

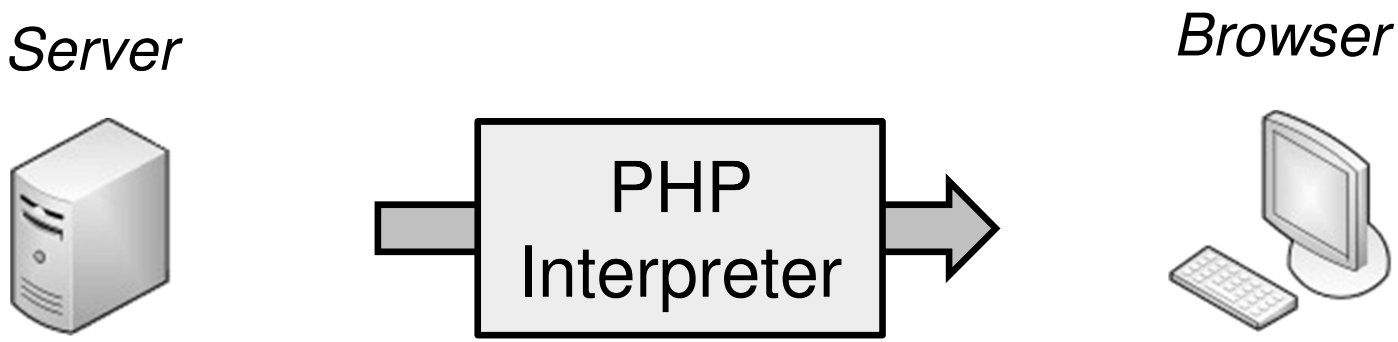
- It is very common to ‘mix’ HTML code with pieces of PHP inside it, as shown in the previous example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Welcome!</title>
</head>
<body>
  <?php echo "Welcome to my page!"; ?>
</body>
</html>
```

A piece of PHP script

Running a PHP File

- For example, the `<?php … ?>` tag in the previous example is replaced by the result of the `echo` command, which is used for showing text output



```
...
<body>
<?php echo "Welcome to
             my page!"; ?>
</body>
...
```

```
...
<body>
Welcome to my page!
</body>
...
```

Short Echo Tags

- It is very common to use the `echo` command to put things into the HTML page
- PHP gives you a quick way to do echo:

  `<?= ...`*Text content to show*`... ?>`

  then you don’t have to type so much, e.g.:

```
...
<body>
<?= "Welcome to my page!" ?>
</body>
...
```

This is equivalent to the code from last slide

The PHP Language

- PHP code is fairly similar to JavaScript code
- PHP variables always start with a `$` sign e.g.:

```
$x = 10; // create variable x
```

- Semi-colons are more important in PHP than JavaScript:

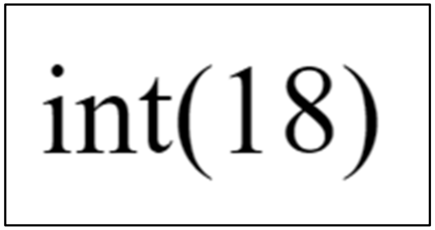
```
$x = $start + 1;
myFunction();
```

Dumping Variable Content

- The `var_dump` function outputs the content of a variable along with its type
- This will be useful when you need to debug a program, e.g.:

```
<?php
$age = 18;
var_dump($age);
?>
```

PHP code runs and the result is sent to the browser



```
int(18)
```

Variable Scope

- You have the typical variable scopes in PHP
  - Local variables,
    i.e. variables created inside any function
  - Global variables,
    i.e. variable created from outside all functions
- To access global variables from within functions, you need to use the `global` keyword, as shown on the next slide

A Simple Example of Scope in PHP

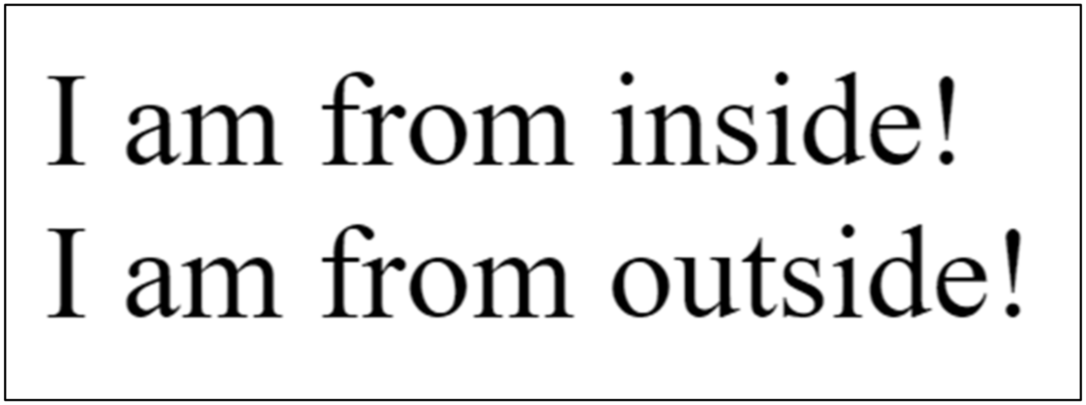
```
<?php
$msg1 = "I am from outside!";
$msg2 = "I am from outside!";

function myFunction() {
    global $msg1;

    $msg1 = "I am from inside!";
    $msg2 = "I am from inside!";
}

myFunction();

echo $msg1;
echo "<br>";
echo $msg2;
?>
```



```
I am from inside!
I am from outside!
```

Operators and Strings

- PHP has all the common operators that you can use in JavaScript
- For strings, you use the period (`.`) operator to concatenate them together, like this:

```
$message = "happy" . " birthday";
```

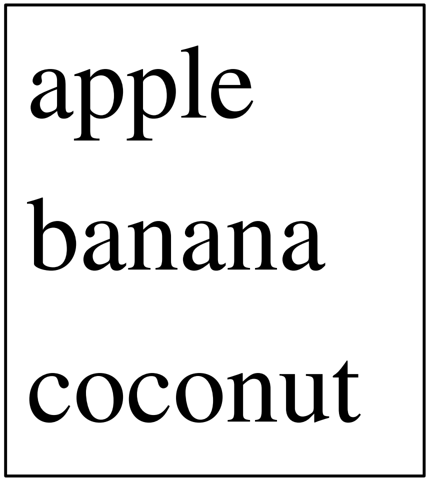
- To get the length of a string, you need to use the `strlen()` function, i.e.:

```
  strlen($message); // return 14
```

Flow Controls

- You can use all the common flow control statements in PHP such as `if`…`else`…, `switch`…, `while`… and `for`…
- You also have `foreach` loop, like this:

```
$veg = ["apple", "banana", "coconut"];
foreach ($veg as $value) {
    echo $value . "<br>";
}
```



```
apple
banana
coconut
```

Functions

- Functions are created the same way as in JavaScript
- You can create named function and anonymous function just like what you would have done in JavaScript

```
<?php
function showMessage() {
    echo "This is from PHP!";
}

showMessage();
?>
```

Arrays

- You can create indexed arrays and associative arrays easily
- Here is an example indexed array:

```
$meals = ["Pancakes","Hotdog", "Curry"];
echo $meals[0]; // output Pancakes
```

- Here is an associative array:

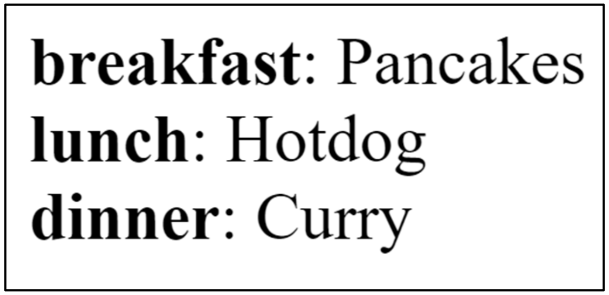
```
$meals = ["breakfast" => "Pancakes",
          "lunch" => "Hotdog",
          "dinner" => "Curry"];
echo $meals["breakfast"]; // output Pancakes
```

Things That You Can Do With Arrays

- You can read the number of things in an array using the `count` function, e.g. :

```
echo count($meals); // 3
```

- A `foreach` loop can read both keys and values from associative arrays, like this:



breakfast: Pancakes
lunch: Hotdog
dinner: Curry

```
foreach ($meals as $meal => $food) {
    echo "<strong>{$meal}</strong>:
          {$food}<br>";
}
```

Variables can be put inside a string too