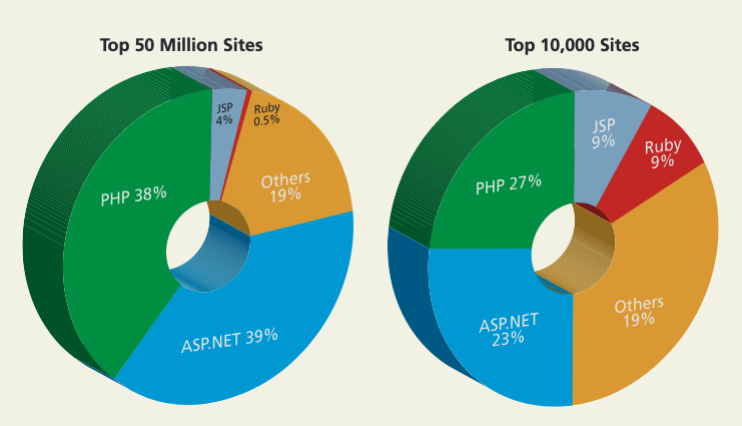
What is server-side development

- Use scripts to dynamically generate content, these scripts run on web server
- Server-side code remains hidden from the client
- Server-side scripts have access to server resources
  - Data storage
  - Database management system

Tech for Server-Side Dev

What are commonly used

- ASP (Active Server Pages) / ASP.NET
- JSP (Java Server Pages)
- Node.js
- Perl
- PHP
- Python
- Ruby on Rails



Developing PHP Locally

In order to develop PHP application we have to install a server that can run PHP and a database management system like

- MySQL
- MariaDB (open-source)

Tech we will be using

Apache or PHP Development Server — HTTP web server software

- Used for running php scripts that receive HTTP request and send responses

MySQL or MariaDB — A database management system

PHP — A server side scripting language "PHP Hypertext Preprocessor"

PHP

PHP Code & HTML markup are written in the same file

- PHP can also be used to generate HTML

Switch between HTML and PHP using PHP tags

```
<!-- HTML CODE GOES HERE -->
<?php /* php code goes in here */ ?>
```

PHP code inside the `<?php ?>` tags is executed, while the HTML markup outside the text is directly outputted to the page

```
<p>This text will be "echoed" directly to the HTML page</p>
<?php
echo "<p>This text is being echoed through the executed PHP script</p>";
?>
```

Variables

- Variables in PHP are dynamically typed
- To declare a variable you must preface the variable name with a dollar symbol `$`
- Whenever you use that variable, you must also include the `$` symbol

```
$count = 42;
```

Data Types

Boolean — A logical true or false value
Integer — Whole numbers
Float — Decimal Numbers
String — Letters
Array — A collection of data of any type
Object — Instances of classes

Strings

Strings can be appended together using concatenate operator `.` symbol

```
$username = "Ricardo";
echo "Hello " . $username; // Outputs Hello Ricardo
```

Escape Strings

`\n` — Line Feed
`\t` — Horizontal Tab
`\\` — Backslash
`\$` — Dollar Sign
`\"` — Double Quote

Constants

Constants never change and always have global scope

- Use `define()`
- Uppercase for constants is a programming convention
- Then use the variable name without quotes or `$`

```
define("DATABASE_LOCAL", "localhost");
echo DATABASE_LOCAL;
```

Program Control

If Else Statement

```
if($hourOfDay > 6 && $hourOfDay < 12) {
    $greetings = "Good Morning";
}
else if($hourOfDay == 12) {
    $greetings = "Good Noon Time";
}
else
    $greetings = "Good Afternoon or Evening"
```

While Loop

```
$count = 0;
while ($count < 10) {
    echo $count;
    $count++;
}
```

For Loop

```
for($count = 0; $count < 100; $count += 5) {
    echo $count;
}
```

Alternate Syntax for Control Structure

```
<? php if($userStatus == "loggedin"): ?>
    <a href="account.php">Account</a>
    <a href="logout.php">Logout</a>
<?php else:?>
```

```
    <a href="login.php">Login</a>
    <a href="register.php">Register</a>
<?php endif;?>
```

Include Files

We can insert file into another
include "somefile.php";
include_once "somefile.php";
require "somefile.php";
require_once "somefile.php";

What is the difference between these four methods of importing files?

Include

Function: Includes and evaluates the specified file
Behaviour on failure: If the file is not found, php throws warning but script will execute
Use case: When file is helpful but not critical to continue

Require

Function: Also includes and evaluates the specified file
Behaviour on failure: if the file is not found, throws an fatal error and stops execution
Use case: When the file is essential to the program (configs, class definitions)

Include_once

Function: works like include but ensures the file is only included once, even if called multiple times
Behaviour on failure: same as include--throws a warning if cant be found but continues

Require_once

Function: works like require, but ensures the file is included only once
Behaviour on failure: Same as require—throws fatal error and stops execution if cant be found

Summary

Statement	Fatal On Error	Prevents Duplicate Inclusion
Include	Warning only	No
Require	Yes	No
Include_once	Warning Only	Yes
Require_once	Yes	Yes

> Use `require_once` for things like
> Function Libs, Configs, Class Declarations

Functions

```
function getNiceTime() {
    return date("H:i:s");
}
echo GetNiceTime();
// or
$output = getNiceTime();
echo $output;
```

A Return Type Declaration Explicitly defines a function's return type This was introduced in v7
Since PHP is dynamically typed, meaning the type is determined at runtime

```
function mustReturnString(): string {
    return "hello";
}
```

Parameters

They're the mechanism that values are passed into functions
To define you must decide:

- How many parameters you want to pass in
- What order they will be passed

```
function getNiceTime($showSeconds) {
    if($showSeconds == true)
        return date("H:i:s");
    else
        return date("H:i");
}
echo getNiceTime(true); // This will print seconds
echo getNiceTime(false); // This will not print seconds
```

Default values

You can set parameter default values, but all the parameters must have defaults if one has default.

```
function getNiceTime($showSeconds=true) {
    if($showSeconds==true)
        return date("H:i:s");
    else
        return date("H:i:");
}
```

Pass By Reference

By default, parameters are pass by value
To pass by reference use the `&` symbol

```
function change(&$arg) {
    $arg += 100;
}
$arg = 15;
change($initial);
echo $initial; // Output is 115
```

Variable Scope Within Functions

All variables defined within a function have function scope, only accessible in the function

Variables defined in the main script are global scope. These variables are not available within the function by default

PHP allows variables with global scope to be accessed within the function with the `global` keyword

```
$count = 56;
function testScope() {
    global $count;
    echo $count;
}
```

Arrays & Superglobals

Arrays

Indexing starts at 0

```
$days = array("Mon", "Tue", "Wed", "Thurs", "Fri");
$days = ["Mon", "Tue", "Wed", "Thrus", "Fri"]; // Alternate Syntax
```

- All arrays in php are associative arrays (key : value)

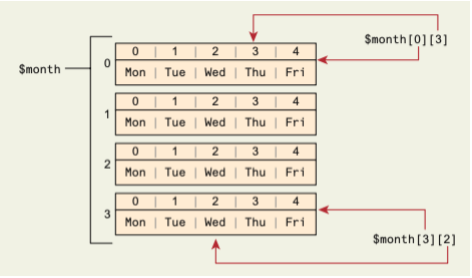
You can use integer and string keys, not necessarily in order

```
$forecast = array("Mon" => 40, "Tue" => 47, "Wed" => 52, "Thrus" => 40, "Fri" => 37);

echo $forecast["Tue"]; // outputs 47
echo $forecast["Thurs"]; // outputs 40
```

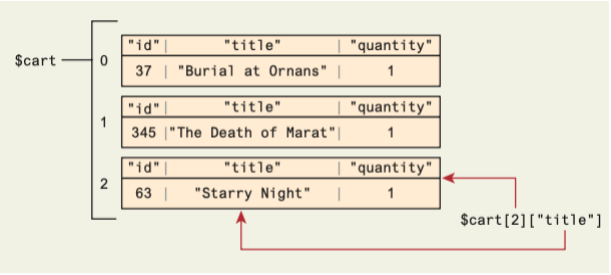
Multidimensional Arrays

```
$month = array(
array("Mon", "Tue", "Wed", "Thrus", "Fri"),
array("Mon", "Tue", "Wed", "Thrus", "Fri"),
array("Mon", "Tue", "Wed", "Thrus", "Fri"),
array("Mon", "Tue", "Wed", "Thrus", "Fri")
);
```



Multidimensional Arrays with Keys

```
$cart = array();
$cart[0] = array("id"=>37, "title"=>"Burial at Ornans", "quantity"=>1);
$cart[1] = array("id"=>345, "title"=>"The Death of Marat", "quantity"=>1);
$cart[2] = array("id"=>63, "title"=>"Starry Night", "quantity"=>1);
```



Iterating Through an Array

While Loop

```
$i = 0;
while($i < count($days)) {
    echo $days[$i] . "<br>";
    i++;
}
```

For Loop

```
for($i = 0; $i < count($days); $i++) {
    echo $days[$i] . "<br>";
}
```

For Each

```
// Iterating through the values of the array $forecast
foreach($forecast as $value) {
    echo $value . "<br>";
}
// Iterating through the values & The keys
foreach($forecast as $key => $value) {
    echo "On".$key." the forecast is ".$value;
}
```

An element can be added by using a key/index that hasnt been used

```
$days[5] = "Sat";
```

Alternative to add to the end

```
$days[] = "Sun";
```

To Delete

```php
unset($days[2]); // Deletes value at index 2, it becomes null
```

Array Sorting

```php
sort($days);
```

More Operations

`array_keys($arr);`

`array_values($arr);`

`array_rand($arr, $num=1);` returns an array of some number of random keys

`array_reverse($arr);`

`array_walk($arr, $callback, $optionalParam);` Call back takes two parameters: $value and $key

`in_array($needle, $haystack);`

Superglobal Arrays

There's predefined associative arrays called superglobal variables which allows the programmer to access HTTP headers, query string params and other common info

Superglobal scope means variables are always in scope and always exist

- can be modified and accessed *without* `global` keyword
  `$GLOBALS` array for storing data that needs superglobal scopoe

`$_COOKIES` array of cookie data passed to page via HTTP request

`$_ENV` array of server environment data

`$_FILES` Array of file items uploaded to the server

`$_GET` array of query string data passed to the server via the URL

`$_POST` array of query string data passed to the server via HTTP header

`$_Request` array containing the contents of `$_GET`, `$_POST` and `$_COOKIES`

`$_SESSION` array that contains session data

`$_SERVER` array containing information about the request and the server

> In this course, `$_Get`, `$_POST` and `$_SERVER` is used mostly

GET

HTML (Client)

```html
<form action="processLogin.php" method="GET">
    Name <input type="text" name="uname"/>
    Pass <input type="text" name="pass"/>
    <input type="submit">
</form>
```

Browser (Client)

Name ricardo/ Pass pw01 Submit

POST

HTML (Client)

```html
<form action="processLogin.php" method="GET">
    Name <input type="text" name="uname"/>
    Pass <input type="text" name="pass"/>
    <input type="submit">
</form>
```

Browser (Client)

Name ricardo/ Pass pw01 Submit

HTTP POST request body:
uname=ricardo&pass=pw01

PHP (Server)

```
echo $_POST["uname"]; // outputs ricardo
echo $_POST["PASS"]; // outputs pw01
```

Determine if any data is sent

use `isset()` function to see if there's any value set for a particular expected key

```
if($_SERVER["REQUEST_METHOD"] == "POST") {
    if(isset($_POST["uname"]) && isset($_POST["pass"])) {
        // handle the posted data
    }
}
```

Sanitizing Query Strings

Just because you're expecting proper query string, it doesnt mean you're going to get one these are the things your program must handle

- Query string parameter doesnt exist
- Query string parameter doesnt contain a value
- Query string parameter value isnt the correct type or is out of acceptable range
- if value is required for a database lookup but provided value doesnt exist in the database