PHP tag -> <?php ?>

echo -> used to output some text.

$ -> used to store a variable

define() -> To define a different type of variable called a constant. (e.g. define(‘NAME’, ‘Yoshi’);)

Tutorial 1-4 work:

<?php

    define('NAME', 'Yoshi');

    // $name = "yoshi";

    $age = 30;

    // $name = 'Mario';

    // $age = 100;

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>My First PHP File</title>

</head>

<body>

    <h1><?php echo 'User Profile Page'; ?></h1>

    <div><?php echo NAME; ?></div>

    <div><?php echo $age ?></div>

</body>

</html>

Variable example:

$stringOne = 'my email is ';

    $stringTwo = 'ros\_ni@outlook.com';

    echo $stringOne.$stringTwo;

The above echo is concatenated by having $stringOne and $stringTwo combined.

The above will output ‘my email is [ros\_ni@outlook.com](mailto:ros_ni@outlook.com)’.



The above will output the value of $name.



The above will output ‘Hey, my name is $name’ because it is not double quoted, so it won’t output the value of $name.

echo "the ninja screamed \"whaaa\"";

What’s done above with the quotes for “whaaa” is done to escape the quotes.



// To replace a certain letter with another letter on selected string. So for the bottom, you are going to replace the M in Mario

    echo str\_replace('M', 'w', $name);

 // Basic operators -> Similar to javascript (\*, /, +, -, +=,

-=, \*=, /=), except for power you use double astricks(\*\*)

    $pi = 3.14;

    $radius = 25;

    // echo $pi \* $radius\*\*2;

    // The bottom will raise the radius by 1, but you cannot do put echo with $radius++ or else the default value will be outputted.

    // Instead, increment first and then echo the $radius.

    $radius++;

    echo $radius;

 // Shorthand operator.

    $age = 20;

    $age -= 10;

    $age \*= 2;

    echo $age;

Arrays

  // Different ways of outputting arrays:

    // Indexed arrays

    $peopleOne = ['shaun', 'cystal', 'ryu'];

    $peopleTwo = array('ken', 'chen-li');

     // When you have to print an array with numbers, it will give an error

because all echoed values need to be a string.

    $ages = [20, 30, 40, 50];

    echo $ages;

 // To print numbers, you have to add print\_r(print readable):

    print\_r($ages);

 // Overriding a value inside the array:

    $ages[1] = 25;

    print\_r($ages);

// Adding more values to an array:

    $ages[] = 60;

    print\_r($ages);

 // To count how many variables are inside an array, we use echo count like so:

    echo count($ages);

// Merging two arrays:

    $peopleThree = array\_merge($peopleOne, $peopleTwo);

    print\_r($peopleThree)

// Associative arrays

    $ninjasOne = ['shaun' => 'black', 'mario' => 'orange', 'luigi' => 'brown'];

    // If you want to print only one of the values inside the arrays, we use echo like so:

    echo $ninjasOne ['mario'];

// multi-dimensional arrays

    $blogs = [

        ['mario party', 'mario', 'lorem', 30],

        ['mario kart cheats', 'toad', 'lorem', 25],

        ['zelda hidden chests', 'link', 'lorem', 50]

    ];

    // To output nested values of a set of arrays amongst arrays, we do the

following:

    print\_r($blogs[0][0])

$blogs = [

        ['title'=>'mario party','author'=>'mario', 'content'=>'lorem', 'likes'=>30],

        ['title'=>'mario kart cheats', 'author'=>'toad', 'content'=>'lorem','likes'=>25],

        ['title'=>'zelda hidden chests', 'author'=>'link', 'content'=>'lorem', 'likes'=>50]

    ];

    // To target an index within a category(such as author, as shown below), we

do the following:

    echo $blogs[2]['author'];

    // We add more data to an array by doing the following:

    $blogs[] = ['title' => 'castle party', 'author'=>'peach', 'content'=>'lorem', 'likes'=>100];

    print\_r($blogs);

    // We can remove data from an array by using the pop function:

    $popped = array\_pop($blogs);

    print\_r($popped);

Loops

Side note: We use period (.) for concatenating.

  // Loops

    // for loop

    $ninjas = ['Rex', 'Haui', 'Yoshi'];

    for($i = 0; $i < count($ninjas); $i++){

        echo $ninjas[$i] . '<br />'; // You have to add a . for concatenating the <br/>.

    }

    // foreach loop

    // The following foreach loop performs the same action as the above for loop, giving you the same result.

    foreach($ninjas as $ninja) {

        echo $ninja . '<br/>';

    }

    $products = [

        ['name' => 'shiny star', 'price' => 20],

        ['name' => 'green shell', 'price' => 10],

        ['name' => 'red shell', 'price' => 15],

        ['name' => 'gold coin', 'price' => 5],

        ['name' => 'lightning bolt', 'price' => 40],

        ['name' => 'banana skin', 'price' => 2]

    ];

    foreach($products as $product){

        echo $product['name'] . ' - ' . $product['price'];

        echo '<br/>';

    }

    $i = 0;

    // While loop

    while($i < count($products)){

        echo $products[$i]['name'];

        echo '<br/>';

        $i++;

    }

<!-- foreach loop example inside html -->

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>PHP Tutorials</title>

</head>

<body>

<h1>Products</h1>

<ul>

<?php foreach($products as $product){?> <!-- Here, we start a tag for the foreach loop.-->

    <h3><?php echo $product['name']; ?></h3>

    <p>$ <?php echo $product['price']; ?> </p>

<?php } ?> <!-- Here, we close the tag for the foreach loop.-->

</ul>

</body>

</html>

// Or (||) and And (&&) examples with foreach loops and if statements

$products = [

        ['name' => 'shiny star', 'price' => 20],

        ['name' => 'green shell', 'price' => 10],

        ['name' => 'red shell', 'price' => 15],

        ['name' => 'gold coin', 'price' => 5],

        ['name' => 'lightning bolt', 'price' => 40],

        ['name' => 'banana skin', 'price' => 2]

    ];

    foreach($products as $product){

            // And percent (&&)

            // if($product['price'] < 15 && $product['price'] > 2){

            //     echo $product['name'] . '<br/>';

            // }

            // Or (||)

            if($product['price'] > 20 || $product['price'] < 10){

                echo $product['name'] . '<br/>';

            }

    }

<!-- The above done on HTML -->

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>PHP Tutorials</title>

</head>

<body>

    <div><ul>

    <?php

    $products = [

        ['name' => 'shiny star', 'price' => 20],

        ['name' => 'green shell', 'price' => 10],

        ['name' => 'red shell', 'price' => 15],

        ['name' => 'gold coin', 'price' => 5],

        ['name' => 'lightning bolt', 'price' => 40],

        ['name' => 'banana skin', 'price' => 2]

    ];

    ?>

    <?php foreach($products as $product){ ?>

    <?php if($product['price'] > 15) {?>

        <li><?php echo $product['name'];?></li>

    <?php }?>

    <?php } ?>

    </ul></div>

</body>

</html>

// Include and require.

// Include basically imports code from the file that

// is exporting the code.

// Require

include('include2.php');

require('include2.php')

include('include3.php');

// // The above include file should give you some sort of

// // error since the file does not exist.

// // However, the echo for this program will still be carried

// // with the execution

require('include3.php');

// However, with require, the echo won't be carried on with

// the execution.

echo 'end of php';

Get and Post

They both send data, but:

* Get sends the data in the URL (Shows the data in the URL Bar)
* Post request sends the data in the request header(hidden)
* <?php
* // GET and POST methods
* // // If the submit button has been set (the submit button has been clicked).
* // if(isset($\_GET['submit'])){
* //     echo $\_GET['email'];
* //     echo $\_GET['title'];
* //     echo $\_GET['ingredients'];
* // }
* if(isset($\_POST['submit'])){
* echo $\_POST['email'];
* echo $\_POST['title'];
* echo $\_POST['ingredients'];
* }
* ?>
* <!DOCTYPE html>
* <html>
* <?php include('header.php'); ?>
* <section class="container grey-text">
* <h4 class="center">Add a pizza</h4>
* <form action="add.php" class="white" method="POST">
* <label>Your Email:</label>
* <input type="text" name="email">
* <label>Pizza Title</label>
* <input type="text" name="title">
* <label>Ingredients (comma seperated):</label>
* <input type="text" name="ingredients">
* <div class="center">
* <input type="submit" name="submit" value="submit" class="btn brand z-depth-0">
* </div>
* </form>
* </section>
* <?php include('footer.php'); ?>
* </html>

<?php

    // HTML chars that we have used below will protect the user from going on

    // to a malicious website which may carry viruses:

    if(isset($\_POST['submit'])){

        echo htmlspecialchars($\_POST['email']);

        echo htmlspecialchars($\_POST['title']);

        echo htmlspecialchars($\_POST['ingredients']);

    }

?>