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']);

    }

?>

<?php

    // Here, we have added validations so that the use can't leave any of the fields

    // empty.

    if(isset($\_POST['submit'])){

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

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

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

    // Check if there is an email entered or if that field's empty

    if(empty($\_POST['email'])){

        echo 'An email is required <br/>';

    } else {

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

    }

     // Check title field

     if(empty($\_POST['title'])){

        echo 'A title is required <br/>';

    } else {

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

    }

     // Check ingredients field

     if(empty($\_POST['ingredients'])){

        echo 'Atleast one ingredient is required <br/>';

    } else {

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

    }

}

?>

MySQL portion

<?php

    // We've created a database using the workbench that you can use by clicking on

    // admin when you're on the xampp control panel and then you can create some data,

    // which is already created and is used by this index.php file.

    // We can use the index.php file to connect to the database and do all sorts of things

    // such as fetching the data and outputting the data in a listed list through the HTML

    // portion.

    // connect to database

    $conn = mysqli\_connect('localhost', 'lolter', 'Test1234', 'ninja\_pizza');

    // check connection

    if(!$conn){

        echo 'Connection error: ' . mysqli\_connect\_error();

    }

    // Write query for all pizzas

    $sql = 'SELECT title, ingredients, id FROM pizzas ORDER BY created\_at';

    // make query and get results

    $result = mysqli\_query($conn, $sql);

    // Fetch the resulting rows as an array

    $pizzas = mysqli\_fetch\_all($result, MYSQLI\_ASSOC);

    // free results from memory

    mysqli\_free\_result  ($result);

    // close connection

    mysqli\_close($conn);

    // print\_r($pizzas);

    // The explode function takes in two arguments (The first one would be some character to look at and seperate it,

    // and the second one would look at the string that needs to be targeted.)

    print\_r(explode(',', $pizzas[0]['ingredients']));

?>

<!DOCTYPE html>

<html>

    <?php include('Project/header.php'); ?>

    <h4 class="center grey-text">Pizzas!</h4>

    <div class="container">

        <div class="row">

        <?php foreach($pizzas as $pizza): ?>

            <div class="col s6 md3">

                <div class="card z-depth-0">

                    <div class="card-content center">

                        <h6><?php echo htmlspecialchars($pizza['title']); ?></h6>

                        <ul>

                        <?php foreach(explode(',', $pizza['ingredients']) as $ing): ?>

                            <li><?php echo htmlspecialchars($ing) ?></li>

                        <?php endforeach ?>

                        </ul>

                    </div>

                    <div class="card-action right-align">

                        <a href="#" class="brand-text">more info</a>

                    </div>

                </div>

            </div>

        <?php endforeach; ?>

        <?php if(count($pizzas)>=3): ?>

            <p> There are 3 or more pizzas</p>

        <?php else: ?>

            <p>There are less than 3 pizzas</p>

        <?php endif; ?>

        </div>

    </div>

    <?php include('Project/footer.php'); ?>

</html>

// Sessions

// Here, we have added a page for a user to enter their name.

// If there is no name, then it will be set to Guest by default.

if(isset($\_POST['submit'])){

    session\_start();

    $\_SESSION['name'] = $\_POST['name'];

    echo $\_SESSION['name'];

    header('Location: index.php');

}

<?php

    session\_start();

    // $\_SESSION['name'] = 'yoshi';

    if($\_SERVER['QUERY\_STRING'] == 'noname'){

        //

        session\_unset();

    }

    $name = $\_SESSION['name'] ?? 'Guest';

?>

// Added cookies to the code.

if(isset($\_POST['submit'])){

    // The 86400 is the seconds before expiration.

    setcookie('gender', $\_POST['gender'], time() + 86400);

    session\_start();

    $\_SESSION['name'] = $\_POST['name'];

    echo $\_SESSION['name'];

    header('Location: index.php');

}

<?php

    session\_start();

    // $\_SESSION['name'] = 'yoshi';

    if($\_SERVER['QUERY\_STRING'] == 'noname'){

        //

        session\_unset();

    }

    $name = $\_SESSION['name'] ?? 'Guest';

    // Get cookie

    $gender = $\_COOKIE['gender'] ?? 'Unknown';

?>

<body class="grey lighten-4">

    <nav class="white z-depth-0">

        <div class="container">

            <a href="/PHP\_Courses/Net\_Ninja/index.php" class="brand-logo brand-text">Ninja Pizza</a>

            <ul id="nav-mobile" class="right hand-on-small-and-down">

                <li class="grey-text">Hello <?php echo htmlspecialchars($name); ?></li>

                <li class="grey-text">(<?php echo htmlspecialchars($gender); ?>)</li>

                <li><a href="add.php" class="btn brand z-depth-0">Add a Pizza</a></li>

            </ul>

        </div>

    </nav>

</body>

 $file = 'quotes.txt';

    // The following code is used to select a file and take an action with it, for example:

    // $handle = fopen(the variable that contains the file ($file in this case), the action to be taken('r' for read))

    $handle = fopen($file, 'r');

    // Read the opened file of the above:

    // echo fread(the variable made to open a file, how many bytes that will be read)

    echo fread($handle, filesize($file));

    // To just read a certain amount of the characters, we do:

    // echo fread(the variable used to open the file, how many characters that need to be read)

    echo fread($handle, 40);

    // Reading a single line:

    echo fgets($handle);

    // Reading a single character:

    echo fgetc($handle);

    // For reading and writing, we add a plus in front of the r in $handle:

    $handle = fopen($file, 'r+');

    fwrite($handle, "\nEverything popular is wrong");

    echo fread($handle, filesize($file));

    // If you want to add a new line at the end of the text, do the following:

    $handle = fopen($file, 'a+');

    fwrite($handle, "\nEverything popular is wrong");

 // Closing a file:

    fclose($handle);

    // Deleting a file

    unlink($file);

The website for more on filesystem openings: <https://www.w3schools.com/php/func_filesystem_fopen.asp>