WEB PROGRAMMING AND APPLICATIONS (Tutorial 6)

Table of Contents

Introduction

PHP is a server scripting language, and a powerful tool for making dynamic and interactive Web pages.

- PHP is an acronym for "PHP: Hypertext Preprocessor"
- PHP is a widely-used, open-source scripting language
- PHP scripts are executed on the server
- PHP is free to download and use

.php File

- PHP files can contain text, HTML, CSS, JavaScript, and PHP code
- PHP code is executed on the server, and the result is returned to the browser
- PHP files have extension ".php"

Basic PHP Syntax

A PHP script can be placed anywhere in the document.

- A PHP script starts with `<?php` and ends with `?>`
- A PHP file normally contains HTML tags, and some PHP scripting code.

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
     echo "Hello World!";
?>

</body>
</html>
```

Variables

In PHP, a variable starts with the `$` sign, followed by the name of the variable:

```
<!DOCTYPE html>
<html>
<body>

<?php
      $x = 5;
      $y = "John";

      echo $x;
      echo "<br>";
      echo $y;
?>

</body>
</html>
```

The PHP echo statement is often used to output data to the screen.

The following example will show how to output text and a variable:

```
$txt = "PHP";
echo "I love " . $txt . "!";
```

PHP Arrays

An array stores multiple values in one single variable:

```
$cars = array("PHP", "JS", "CSS");
```

In PHP, there are three types of arrays:

- **Indexed arrays** - Arrays with a numeric index
- **Associative arrays** - Arrays with named keys
- **Multidimensional arrays** - Arrays containing one or more arrays

Indexed Arrays

To access an array item you can refer to the index number.

```
$cars = array("Volvo", "BMW", "Toyota");
echo $cars[0];
```

To change the value of an array item, use the index number:

```
$cars = array("Volvo", "BMW", "Toyota");
$cars[1] = "Ford";
```

To loop through and print all the values of an indexed array, you could use a foreach loop, like this:

```php
$cars = array("Volvo", "BMW", "Toyota");
foreach ($cars as $x) {
  echo "$x <br>";
}
```

Associative Arrays

To access an array item, you can refer to the key name.

```php
$car = array("brand"=>"Ford", "model"=>"Mustang", "year"=>1964);
echo $car["model"];
```

To change the value of an array item, use the key name:

```php
$car = array("brand"=>"Ford", "model"=>"Mustang", "year"=>1964);
$car["year"] = 2024;
```

To loop through and print all the values of an indexed array, you could use a foreach loop, like this:

```php
$car = array("brand"=>"Ford", "model"=>"Mustang", "year"=>1964);

foreach ($car as $x => $y) {
  echo "$x: $y <br>";
}
```

Multidimensional Arrays

A multidimensional array is an array containing one or more arrays.

PHP supports multidimensional arrays that are two, three, four, five, or more levels deep. However, arrays more than three levels deep are hard to manage for most people.

A two-dimensional array is an array of arrays (a three-dimensional array is an array of arrays of arrays).

Name	Stock	Sold
Volvo	22	18
BMW	15	13
Saab	5	2
Land Rover	17	15

We can store the data from the table above in a two-dimensional array, like this:

```
$cars = array (
  array("Volvo",22,18),
  array("BMW",15,13),
  array("Saab",5,2),
  array("Land Rover",17,15)
);
```

Loops

Loops are used to execute the same block of code again and again, as long as a certain condition is true.

In PHP, we have the following loop types:

- `while` - loops through a block of code as long as the specified condition is true
- `do...while` - loops through a block of code once, and then repeats the loop as long as the specified condition is true
- `for` - loops through a block of code a specified number of times
- `foreach` - loops through a block of code for each element in an array

```
$i = 1;
while ($i < 6) {
  echo $i;
  $i++;
}
```

```php
$i = 1;

do {
  echo $i;
  $i++;
} while ($i < 6);
```

```php
for ($x = 0; $x <= 10; $x++) {
  echo "The number is: $x <br>";
}
```

```php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $x) {
  echo "$x <br>";
}
```

Form Handling

The PHP superglobals `$_GET` and `$_POST` are used to collect form-data.

```
<html>
<body>

<form action="welcome.php" method="POST">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

The "welcome.php" looks like this:

```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo $_POST["email"]; ?>

</body>
</html>
```

Content-type

application/json

```
let name = document.querySelector("#name").value
let comment = document.querySelector("#comment").value
let type = document.querySelector("#type").value
fetch("ex3-add-comment.php", {
    method: "post",
    headers: {
```

```
        "content-type": "application/json"
    },
    body: JSON.stringify({
        name, comment, type
    })
})
    .then(response => {
        if (!response.ok) throw new Error("failed to add comment")

        return response.json()
    })
    .then(jsonObj => {
        ...
    })
    .catch(error => {
        alert(error.message)
    })
```

```
<?php
    header('Content-Type: application/json; charset=utf-8');

    $comments = file_get_contents('comments.txt');
    $comments = explode("\n", $comments);
    $comments = array_filter($comments); // Remove empty lines

    $result = array();
    foreach (array_reverse($comments) as $comment) {
        list($name, $comment, $type, $timestamp) = explode('|', $comment);
        $item = array(
            "name"=> $name,
            "comment"=> $comment,
            "type" => $type,
            "timestamp"=> $timestamp
        );
        $result[] = $item;
    }

    echo json_encode($result);
```

```
fetch(BACKEND_API_URL, {
    method: 'post',
    headers: {
        'content-type': 'application/x-www-form-urlencoded'
    },
    body: new URLSearchParams({
        num1, num2, operator: operator
    }).toString()
})
    .then(response => {
        if(!response.ok) throw new Error('failed to call api')

        return response.text()
    })
    .then(result => {
        document.querySelector('div[class="alert alert-success"]').innerText = result
    })
    .catch(error => {
        alert(error.message)
    })
```

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    if (isset($_POST['num1']) && isset($_POST['num2']) && isset($_POST['operator'])) {
        $num1 = $_POST['num1'];
        $num2 = $_POST['num2'];
        $operator = $_POST['operator'];

        switch ($operator) {
            case '+':
                $result = $num1 + $num2;
                break;
            case '-':
                $result = $num1 - $num2;
                break;
            case 'x':
                $result = $num1 * $num2;
                break;
            case '/':
                if ($num2 != 0) {
                    $result = $num1 / $num2;
                } else {
                    $result = "Invalid params!";
                }
                break;
            default:
```

```
                $result = "Invalid params!";
                break;
        }

        echo $result;
    }
}
?>
```

PHP and JSON

JSON stands for JavaScript Object Notation, and is a syntax for storing and exchanging data. Since the JSON format is a text-based format, it can easily be sent to and from a server, and used as a data format by any programming language.

PHP has some built-in functions to handle JSON.

First, we will look at the following two functions:

- json_encode()
- json_decode()

JSON encode

```
<?php
$age = array("Peter"=>35, "Ben"=>37, "Joe"=>43);

echo json_encode($age);
?>
```

➔ {"Peter":35,"Ben":37,"Joe":43}

```php
<?php
$cars = array("Volvo", "BMW", "Toyota");

echo json_encode($cars);
?>
```

➔ ["Volvo","BMW","Toyota"]

JSON decode

This example shows how to access the values from a PHP object:

```php
<?php
$jsonobj = '{"Peter":35,"Ben":37,"Joe":43}';

$obj = json_decode($jsonobj);

echo $obj->Peter;
echo $obj->Ben;
echo $obj->Joe;
?>
```

This example shows how to access the values from a PHP associative array:

```php
<?php
$jsonobj = '{"Peter":35,"Ben":37,"Joe":43}';

$arr = json_decode($jsonobj, true);

echo $arr["Peter"];
echo $arr["Ben"];
echo $arr["Joe"];
?>
```

This example shows how to loop through the values of a PHP object:

```php
<?php
$jsonobj = '{"Peter":35,"Ben":37,"Joe":43}';

$obj = json_decode($jsonobj);

foreach($obj as $key => $value) {
  echo $key . " => " . $value . "<br>";
}
?>
```

This example shows how to loop through the values of a PHP associative array:

```php
<?php
$jsonobj = '{"Peter":35,"Ben":37,"Joe":43}';

$arr = json_decode($jsonobj, true);

foreach($arr as $key => $value) {
  echo $key . " => " . $value . "<br>";
}
?>
```

References

https://www.w3schools.com/php/default.asp