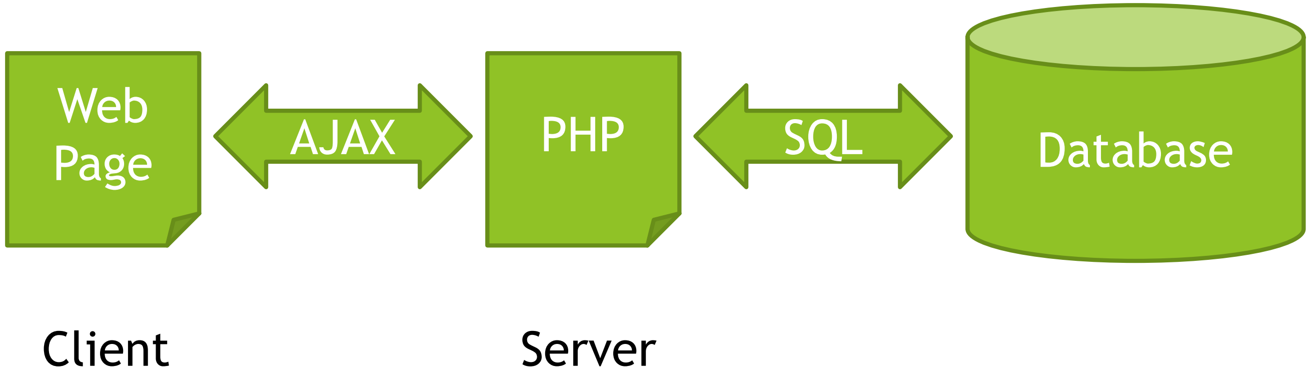
HTML 5 and JavaScript Game Programming

Week 9

Mario.Soflano@uws.ac.uk

Client-Server System Architecture



Web Page

AJAX

PHP

SQL

Database

Client

Server

Userid	Score
User 1	5
User 2	8
User 3	2

The client send a request and data to the server.

PHP is a back-end / server-side web programming language. PHP handles request and data from the client, process it and send it to the database

The Structured Query Language (SQL) is a database programming language to manipulate data in database

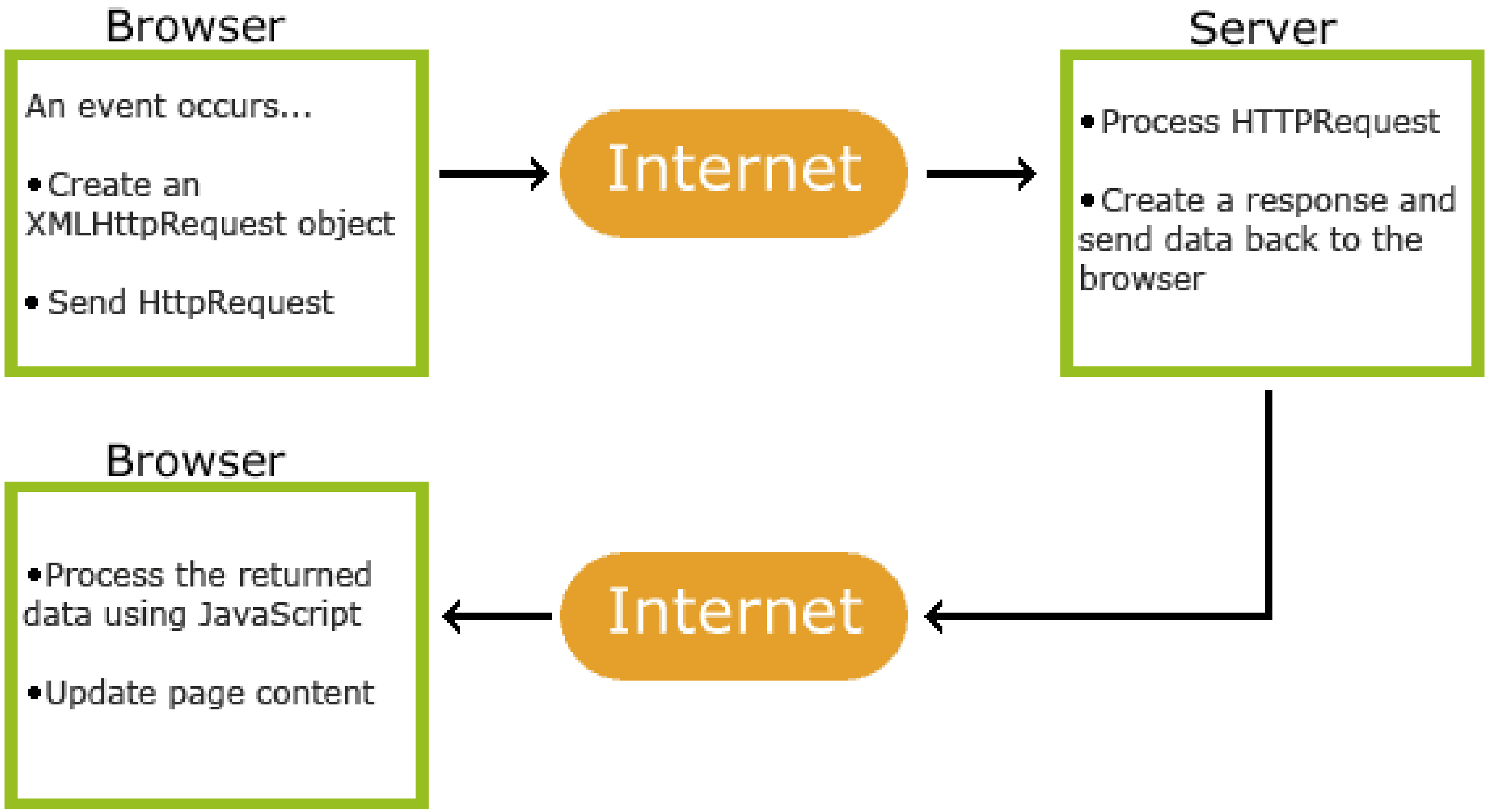
Data in database can be presented in columns and rows

PHP ‘executes’ the SQL statement

AJAX

- AJAX = Asynchronous JavaScript and XML
- AJAX is a technique for creating fast and dynamic web pages.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.
- Classic web pages, (which do not use AJAX) must reload the entire page if the content should change.
- Examples of applications using AJAX: Google Maps, Gmail, YouTube, and Facebook tabs.
- AJAX is based on internet standards, and uses a combination of:
    - XMLHttpRequest object (to exchange data asynchronously with a server)
    - JavaScript/DOM (to display/interact with the information)
    - CSS (to style the data)
    - XML (often used as the format for transferring data)

AJAX



Browser

An event occurs...

- Create an XMLHttpRequest object
- Send HttpRequest

Internet

Server

- Process HTTPRequest
- Create a response and send data back to the browser

Internet

Browser

- Process the returned data using JavaScript
- Update page content

AJAX

(Within Javascript block code)

```
if (window.XMLHttpRequest) {
            // code for IE7+, Firefox, Chrome, Opera, Safari
            xmlhttp = new XMLHttpRequest();
     } else {
            // code for IE6, IE5
             xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
      }
      xmlhttp.onreadystatechange = function() {
            if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                          document.getElementById("txtHint").innerHTML = xmlhttp.responseText;
             }
      };

}

//Use GETmethod

     xmlhttp.open("GET",“thePHPfile.php?thePassVar1=‘testValue1’&thePassVar2=‘testValue2’");
     xmlhttp.send();
```

AJAX

```
xmlhttp = new XMLHttpRequest();
xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
```

Are to check the browser and use the relevant XMLHTTP Block

```
xmlhttp.onreadystatechange
```

Is to check any response from the server

```
xmlhttp.open("GET",“thePHPfile.php?thePassVar1=‘testValue1’&thePassVar2=‘testValue2’");
```

xmlhttp.open(“GET”, “getuser.php?q=” + str, true); means the method is GET, the target file is “thePHPfile.php”, the passing variable are “thePassVar1” with the value of ‘testValue1’ and “thePassVar2” with the value of ‘testValue2’

AJAX

//Use POST method

```
//Format the data into a Javascript Object
var obj = { 'user': ‘user1’, 'thescore': 5 };

xmlhttp.open('POST', 'thePHPfile.php'); // use POST method, the target file is 'thePHPfile.php‘

xmlhttp.setRequestHeader("Content-Type", 'application/x-www-form-urlencoded');

//Convert Javascript object to JSON

jsonData = JSON.stringify(obj);

xmlhttp.send('jsonPassVar=' + jsonData);
```

http://www.w3schools.com/tags/ref_httpmethods.asp

(comparing the GET and POST method)…however in AJAX, the difference in the process of both methods is not that significant

PHP - Intro

- PHP is a server scripting language
- PHP is free and a good alternative to Microsoft’s ASP
- The default file extension for PHP files is ".php“
- In PHP, a variable starts with the $ sign, followed by the name of the variable
- PHP start:

```
<?php
```

- PHP end:

```
?>
```

http://www.w3schools.com/php/php_intro.asp

PHP - Example

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

<?php
// This is a single-line comment

# This is also a single-line comment

/*
This is a multiple-lines comment block
that spans over multiple
lines
*/

// You can also use comments to leave out parts of a code line
$x = 5 /* + 15 */ + 5;
echo $x;
?>

</body>
</html>
```

PHP - Conditions

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

<?php
    $firstNumber = 3;
    $secondNumber = 7;
    if($firstNumber > $secondNumber ) {
        echo “the first number is higher”;
    } else {
        echo “the second number is higher”;
    };
?>

</body>
</html>
```

PHP - Conditions

```
<?php
    $favcolor = "red";

    switch ($favcolor) {
        case "red":
            echo "Your favorite color is red!";
            break;
        case "blue":
            echo "Your favorite color is blue!";
            break;
        case "green":
            echo "Your favorite color is green!";
            break;
        default:
            echo "Your favorite color is neither red, blue, nor green!";
    }
?>
```

PHP - Loops

//Basic For loops

```
for (init counter; test counter; increment counter) {
        code to be executed;
    }
```

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

?>
```

//Loop for array

```
foreach ($array as $value) {
        code to be executed;
    }
```

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

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

PHP – Handling Ajax

```
//Handling the data sent by GET method

$thePassValue = $_GET['q'];


// Handling the JSON data sent by POST method

      //First, decode the JSON
```

json_decode (string $json [, bool $assoc = false [, int $depth = 512 [, int $options = 0]]])

string $json: The json string being decoded.

bool $assoc: When **TRUE**, returned objects will be converted into associative arrays.

int $depth: User specified recursion depth

int $options: Bitmask of JSON decode options. Currently only **JSON_BIGINT_AS_STRING** is supported (default is to cast large integers as floats)

```
$request_json = json_decode($_POST['jsonPassVar '], true);


      //Second, access the particular data

      $theuser = $request_json['user'];

      $thescore = $request_json['thescore'];
```

Structured Query Language (SQL)

Extract data from database

//retrieve all data from *table_name*

SELECT * FROM *table_name*;

SELECT * FROM thescore_tbl;

//retrieve data from particular column(s)

SELECT *column_name*,*column_name* FROM *table_name*;

SELECT userid FROM thescore_tbl;

// retrieve data from particular column(s) with a condition

SELECT *column_name*,*column_name* FROM *table_name* WHERE *column_name operator value*;

SELECT userid FROM thescore_tbl WHERE thescore >3

Structured Query Language (SQL)

Insert data to database

//insert data to *table_name*

INSERT INTO *table_name* (*column1*,*column2*,*column3*,...) VALUES (*value1*,*value2*,*value3*,...);

INSERT INTO thescore_tbl(userid, thescore) VALUES (‘Mario’, ‘7’);

Update data in database

UPDATE *table_name* SET *column1*=*value1*,*column2*=*value2*,... WHERE *some_column*=*some_value*;

UPDATE thescore_tbl SET thescore = 4 WHERE userid = ‘Mario’;

Structured Query Language (SQL)

Executing SQL in PHP

```
//First, connect to database (host, username, password)
$con = mysql_connect('localhost','root','');
if (!$con) {
    echo "Failed to make connection.";
    exit;
}

//Second, select a database
$db = mysql_select_db('phaserdb');

//Third, ‘execute’ the SQL input statement
$sql = "INSERT INTO score_tbl (userid,userscore) VALUES ('".$theuser."',$thescore)";
$query = mysql_query($sql);
```

Alternative Solutions

- Using JQuery

  https://www.youtube.com/watch?v=TR0gkGbMwW0

- Using NodeJS

  https://www.youtube.com/watch?v=e8ZLfcHxrD8

References

http://www.w3schools.com/sql/

http://www.w3schools.com/Ajax/ajax_intro.asp

http://www.w3schools.com/php/

http://www.w3schools.com/tags/ref_httpmethods.asp

http://php.net/manual/en/