PHP continue ...Java Script.. Ajax

Connectivity with MySQL

MySQL Connectivity

- Ways to connected to mysql database?
  - Using MySQLi
  - Using PDO String

MySQL Connectivity

- What is DSN
  - A DSN is basically a string of options that tell PDO which driver to use, and the connection details.
- Connectivity with database
  - Example

```
<?php
$db = new PDO('mysql:host=localhost;dbname=testdb;charset=utf8mb4', 'username', 'password');
?>
 <?php
                $db = new PDO('mysql:host=localhost;dbname=testdb;charset=utf8mb4', 'username', 'password',
array(PDO::ATTR_EMULATE_PREPARES => false,
                                PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION));
?>
<?php
                $db = new PDO('mysql:host=localhost;dbname=testdb;charset=utf8mb4', 'username', 'password');
                $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
                $db->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
?>
```

MySQL Connectivity

- Running Simple Select Statements
  - **Example**

```
<?php
        foreach($db->query('SELECT * FROM table') as $row) {
                echo $row['field1'].' '.$row['field2']; //etc...}
?>
<?php
        $stmt = $db->query('SELECT * FROM table');

        while($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
                echo $row['field1'].' '.$row['field2']; //etc...}
?>
<?php
        $stmt = $db->query('SELECT * FROM table');
        $results = $stmt->fetchAll(PDO::FETCH_ASSOC);
        //use $results
?>
```

MySQL Connectivity

- Running Simple INSERT, UPDATE, or DELETE statements
  - **Example**

```
<?php
        $affected_rows = $db->exec("UPDATE table SET field='value'");
        echo $affected_rows.' were affected‘ ?>
```

- INSERT, DELETE, UPDATE Prepared Queries

```
<?php
        $stmt = $db->prepare("INSERT INTO table(field1,field2,field3,field4,field5) VALUES(:field1,:field2,:field3,:field4,:field5)");
        $stmt->execute(array(':field1' => $field1, ':field2' => $field2, ':field3' => $field3, ':field4' => $field4, ':field5' => $field5));
        $affected_rows = $stmt->rowCount();
?>
<?php
        $stmt = $db->prepare("UPDATE table SET name=? WHERE id=?");
        $stmt->execute(array($name, $id));
        $affected_rows = $stmt->rowCount();
?>
<?php
        $stmt = $db->prepare("DELETE FROM table WHERE id=:id");
        $stmt->bindValue(':id', $id, PDO::PARAM_STR);
        $stmt->execute();
        $affected_rows = $stmt->rowCount();
?>
```

Javascript

Javascript

What is it

Javascript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

Javascript

Key points regarding javascript

- JavaScript is a lightweight, interpreted programming language.
- Designed for creating network-centric applications.
- Complementary to and integrated with Java.
- Complementary to and integrated with HTML.
- Open and cross-platform

Javascript

What it is Client-Side script?

- Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser.

Javascript

Advantages of Javascript

- Less server interaction
- Immediate feedback to the visitors
- Increased interactivity
- Richer interfaces

Limitations of JavaScript

- Client-side JavaScript does not allow the reading or writing of files. This has been kept for security reason.
- JavaScript cannot be used for networking applications because there is no such support available.
- JavaScript doesn't have any multithreading or multiprocessor capabilities.

`

Javascript(Variables)

Example:

```
<script type="text/javascript">
  <!--
    var money;
    var name;
  //-->
</script>

<script type="text/javascript">
  <!--
    var money, name;
  //-->
</script>
```

Javascript(Operators)

Supported operators

- Arithmetic Operators
- Comparison Operators
- Logical (or Relational) Operators
- Assignment Operators
- Conditional (or ternary) Operators

Javascript

If … else

- Syntax

```
if (expression 1){
  Statement(s) to be executed if expression 1 is true
}

else if (expression 2){
  Statement(s) to be executed if expression 2 is true
}

else if (expression 3){
  Statement(s) to be executed if expression 3 is true
}

else{
  Statement(s) to be executed if no expression is true
}
```

Javascript(If … else)

Example

```
<html>
  <body>

    <script type="text/javascript">
      <!--
        var book = "maths";
        if( book == "history" ){
          document.write("<b>History Book</b>");
        }

        else if( book == "maths" ){
          document.write("<b>Maths Book</b>");
        }

        else if( book == "economics" ){
          document.write("<b>Economics Book</b>");
        }

        else{
          document.write("<b>Unknown Book</b>");
        }
      //-->
    </script>

    <p>Set the variable to different value and then try...</p>
  </body>
<html>
```

Javascript

Loop

- Syntax
  - For Loop

    for (initialization; test condition; iteration statement){

      Statement(s) to be executed if test condition is true

    }
  - While loop

    while (expression){

      Statement(s) to be executed if expression is true

    }

Javascript(Loop)

Example: For Loop

```
<html>
  <body>
    <script type="text/javascript">
      <!--
        var count;
        document.write("Starting Loop" + "<br />");
        for(count = 0; count < 10; count++){
          document.write("Current Count : " + count );
          document.write("<br />");
        }
        document.write("Loop stopped!");
      //-->
    </script>
    <p>Set the variable to different value and then try...</p>
  </body>
</html>
```

Example: while Loop

```
<html>
  <body>
    <script type="text/javascript">
      <!--
        var count = 0;
        document.write("Starting Loop ");
        while (count < 10){
          document.write("Current Count : " + count + "<br />");
          count++;
        }
        document.write("Loop stopped!");
      //-->
    </script>
    <p>Set the variable to different value and then try...</p>
  </body>
</html>
```

Javascript

Function

- Syntax
  - Basic

```
<script type="text/javascript">
<!--
   function functionname(parameter-list)
   {
      statements
   }
//-->
```

Javascript(Functions)

Example:

```
<html>
  <head>

    <script type="text/javascript">
      function concatenate(first, last)
      {
        var full;
        full = first + last;
        return full;
      }

      function secondFunction()
      {
        var result;
        result = concatenate('Zara', 'Ali');
        document.write (result );
      }
    </script>

  </head>

<body>
    <p>Click the following button to call the function</p>

    <form>
      <input type="button" onclick="secondFunction()" value="Call Function">
    </form>

    <p>Use different parameters inside the function and then try...</p>

  </body>
</html>
```

Javascript(Form validation)

Example:

```
<html>
<head>
<title>Untitled Document</title>
<script type="text/javascript">
  <!--

    // Form validation code will come here.
    function validate()
    {

      if( document.fromSignin.txtuname.value == "" )
      {
        alert( "Please provide your name!" );
        document.fromSignin.txtuname.focus() ;
        return false;
      }
                                    return true;
                }
                                    -->
</script>
</head>
```

```
<body>
<form name="fromSignin" method="post" action="response.php" onSubmit="return(validate());">
<p>
  <label for="textfield">User Name:</label>
  <input type="text" name="txtuname" id="txtuname">
</p>
<p>
  <label for="password">Password:</label>
  <input type="password" name="password" id="password">
</p>
                <input type="submit" value="Sign-In">
                </form>
<p> </p>
</body>
</html>
```

AJAX

AJAX

What is it

AJAX stands for **A**synchronous **Ja**vaScript and **X**ML. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script.

AJAX

Key Points for AJAX

- Ajax uses XHTML for content, CSS for presentation, along with Document Object Model and JavaScript for dynamic content display.
- Conventional web applications transmit information to and from the server using synchronous requests. It means you fill out a form, hit submit, and get directed to a new page with new information from the server.
- With AJAX, when you hit submit, JavaScript will make a request to the server, interpret the results, and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server.
- XML is commonly used as the format for receiving server data, although any format, including plain text, can be used.
- AJAX is a web browser technology independent of web server software.
- A user can continue to use the application while the client program requests information from the server in the background.
- Intuitive and natural user interaction. Clicking is not required, mouse movement is a sufficient event trigger.
- Data-driven as opposed to page-driven.

AJAX

Examples

- Google Maps
- Google Suggest
- Gmail
- Yahoo Maps (new)

AJAX

Steps of AJAX Operation

- A client event occurs.
- An XMLHttpRequest object is created.
- The XMLHttpRequest object is configured.
- The XMLHttpRequest object makes an asynchronous request to the Webserver.
- The Webserver returns the result containing XML document.
- The XMLHttpRequest object calls the callback() function and processes the result.
- The HTML DOM is updated.

AJAX

Example:

```
<html>
  <head>
    <style>
      span {
        color: green;
      }
    </style>
    <script>
      function showHint(str) {
        if (str.length == 0) {
          document.getElementById("txtHint").innerHTML = "";
          return;
        }else {
          var xmlhttp = new XMLHttpRequest();

          xmlhttp.onreadystatechange = function() {
            if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
              document.getElementById("txtHint").innerHTML = xmlhttp.responseText;
            }
          }
          xmlhttp.open("GET", "php_ajax.php?q=" + str, true);
          xmlhttp.send();
        }
      }
    </script>
  </head>
```

```
<body>
    <p><b>Search your favourite tutorials:</b></p>
     <form>
       <input type = "text" onkeyup = "showHint(this.value)">
    </form>
    <p>Entered Course name: <span id="txtHint"></span></p>
  </body>
</html>
```

AJAX

Example: (php_ajax.php)

```php
<?php
// Array with names
$a[] = "Android";
$a[] = "B programming language";
$a[] = "C programming language";
$a[] = "D programming language";
$a[] = "euphoria";
$a[] = "F#";
$a[] = "GWT";
$a[] = "HTML5";
$a[] = "ibatis";
$a[] = "Java";
$a[] = "K programming language";
$a[] = "Lisp";
$a[] = "Microsoft technologies";
$a[] = "Networking";
$a[] = "Open Source";
$a[] = "Prototype";
$a[] = "QC";
$a[] = "Restful web services";
$a[] = "Scrum";
$a[] = "Testing";
$a[] = "UML";
$a[] = "VB Script";
$a[] = "Web Technologies";
$a[] = "Xerox Technology";
$a[] = "YQL";
$a[] = "ZOPL";

$q = $_REQUEST["q"];
  $hint = "";

  if ($q !== "") {
    $q = strtolower($q);
    $len = strlen($q);

    foreach($a as $name) {

      if (stristr($q, substr($name, 0, $len))) {
        if ($hint === "") {
          $hint = $name;
        }else {
          $hint .= ", $name";
        }
      }
    }
  }
  echo $hint === "" ? "Please enter a valid course name" : $hint;
?>
```