Programación Web (2015-2016)

Grado en Ingeniería Informática

Universidad de Granada

Práctica 2: Adaptación de la web de la tienda de música

Mª Cristina Heredia Gómez

5 de junio de 2016

Descripción

A lo largo de esta práctica se han ido completando ejercicios que, mediante el uso de HTML, CSS, PHP, Javasscript y MySQL han permitido dotar a la tienda de música de la práctica 1, de algunas funcionalidades, como identificación de usuarios, creación y validación de formularios o inserción, consulta y borrado de discos, comentarios y canciones. Todo ello habría sido más complicado sin la ayuda de `http://www.w3schools.com/`. Veamos como se han ido reolvendo cada uno de los ejercicios:

Ejercicio 1

En este ejercicio se nos plantea que, a partir del formulario de incripción de la práctica anterior se dieran de alta clientes en la base de datos.
Para resolverlo, simplemente tenemos que procesar el formulario una vez que el usuario pulse el botón de enviar. Para ello indicamos en el $< form >$ la acción de llamar a **procesar.php**, que será el fichero donde procesaremos los datos introducidos.

```
<section id="formulario">
<section>
<form action="procesar.php"  name="f1"  onsubmit="return(validateForm1());" method="post">
```

Luego, solo queda procesar la información, para ello tenemos un archivo de configuración que contiene la información necesaria para conectarse a la base de datos y que es utilizado por **configuracion.php** para realizar la conexión a la base de datos de clientes e insertar la información mediante una consulta SQL:

```
    // prepare sql and bind parameters
    $sql_query = $conexion->prepare("INSERT INTO Clientes (Nombre,Apellidos,Nombre_Usuario,Clave,Direccion,
Ciudad,Codigo_Postal,Provincia,Correo,DNI,VISA,Observaciones,Suscripcion)
    VALUES (:Nombre,:Apellidos,:Nombre_Usuario,:Clave,:Direccion,:Ciudad,:Codigo_Postal,:Provincia,
    :Correo,:DNI,:VISA,:Observaciones,:Suscripcion)");
```

Ejercicio 2

Se demanda modificar la página principal de la web para permitir identificar al cliente y abrir de esa forma una sesión que será cerrada cuando el cliente lo solicite. Para resolverlo, en el formulario de identificación de usuario indicamos que cuando se pulse en acceder, iremos a **login.php**, donde vamos a manejar todo lo relativo al inicio de sesión.
Una vez en textitlogin.php comprobaremos si la clave y usuario se corresponden y si es así, se inicia una sesión y se vuelve a la página de inicio. Para Comprobar si clave y usuario se corresponden:

```
function isClient($client,$password){

try {
    $conexion = new PDO(DB_DSN, DB_USUARIO, DB_CONTRASENIA);
    $sql_query =$conexion->prepare("SELECT Clave FROM Clientes WHERE Nombre_Usuario='$client'");
```

```
    $sql_query->execute();

    $resultado="";
    while($row=$sql_query->fetch(PDO::FETCH_OBJ)) {
        /*its getting data in line.And its an object*/
        $resultado= $row->Clave;
    }

}catch(PDOException $e)
    {
        echo "Error: " . $e->getMessage();
    }
    $conexion = null;

    if($resultado===$password) return true;
    else return false;

}
```

Para establecer variable de sesión:

```
    if(isClient($user,$pass))  // user is in client's dataset
    {

        $_SESSION['use']=$user;
    }
    else
    {
        echo "invalid UserName or Password";
    }
```

Para volver a la página principal:

```
    header("Location:index.php");
```

Una vez en la página principal, pondremos

para que la sesión se mantenga activa hasta que el usuario decida cerrarla, le damos la bienvenida con un *echo* de su nombre y le mostramos un botón de Salir que le permita cerrar la sesión cuando lo desee.

```
    if(isset($_SESSION['use'])) { // If session is not set that redirect to Login Page
                // header("Location:index.php");

                echo " Login Success ";
                echo ' ';
```

```
        echo "hola, ".$_SESSION['use'];
    if($_SESSION['use']==='Admin') header("Location:insertDiskForm.php");
    //echo "<a href='logout.php'> Logout</a> ";
?>
    <form id="login" action="index.php" method="post" name="f2">
      <input type = 'submit' id = 'outbutton1' name='logout' value = 'Salir' />
    </form>

    <?php

    if(isset($_POST['logout']))
        header("Location:logout.php");
    }else{ //show identification form
```

Finalmente, **logout.php** cerramos la sesión con un $session\_destoy()$ y redirigimos a la página inicial.

Ejercicio 3

Crear un formulario para que un usuario identificado pueda comentar discos y que éstos se muestren en los detalles del mismo, apareciendo en los comentarios tanto la fecha, como el usario que lo puso y el comentario.
Este ha sido el ejercicio más laborioso de todos, en mi opinión. Para resolverlo, he creado tres clases: **Disco,Comentario y DataObject** donde Disco y Comentario heredan de DataObject, (que según hemos visto en clase,es lo conveniente). En la clase Disco, se implementan los métodos para obtener un disco dado su nombre, obtener todos los discos de una sección, obtener los comentarios de un disco dado su nombre como parámetro e insertar un nuevo Disco en la base de datos.
En la clase DataObject se implementan funcionalidades generales,como conectar/desconectar de la base de datos, y en **Comentario** se implementa la opción de insertar un comentario en la base de datos, dado un disco,un usuario y un comentario. una vez que el usuario está logeado, al seleccionar un disco le aparecerá un formulario para comentarlo si lo desea, así como los comentarios asociados a ese Disco.
Formulario para comentar disco:

```
            name="f2" onsubmit="return(validateForm2());">
<?php        if(isset($_SESSION['use'])){
?>
        <fieldset id="campoComentaForm">
                <text>¡pon tu comentario!</text>
                Usuario:
                <input type="text" name="usuario" value="<?php if (isset($_SESSION['use']))
                {echo $_SESSION['use']; }?>" required >
                Comentario:
                <textarea rows="3" name="comentario" value=""  placeholder="Comentario"
                                required></textarea>
                <input type = "submit" id = "button1" value = "Comentar"/>
```

```
        </fieldset>
        <?php
        }else  echo "identificate para comentar!";
```

Obtener y mostrar todos los comentarios para un nombre de disco dado:

```
$comentariosDisco=$Objetodisco->obtenerComentariosDisco($nombreDisco);
if(count($comentariosDisco)>0){
?>
        <text>Comentarios : </text>
<section id="comments">
        <?php
                foreach ($comentariosDisco as $row){
        ?>
        <section id="commentInd">
                <p id="cabezeraCom"><?php echo $row['Nombre_Usuario'] ." , ". $row['fecha']. ":

        <section id="commentBox">
                <p><em><?php echo $row['Comentario']; ?></em></p>
        </section>
        </section>
</section>

<?php
}}
?>
```

Definimos en el CSS todas las cuestiones de estilo. Para procesar el comentario introducido se llama a **procesarComentario.php** pasándole el id del disco que queremos comentar, e insertará el comentario en la base de datos.

Ejercicio 4

Para realizar la validación de todos los formularios en javascript, he usado en todos los casos expresiones regulares, pues me parecen lo más cómodo. En caso de que no se pueda hacer un match entre la expresión regular considerada y el valor introducido, se muestra un **alert** indicando un error en dicho campo.
Además, también he usado el valor **required** en aquellos elementos del html que son obligatorios , y he usado algunas funciones **filter()** de PHP para sanitizar los valores introducidos por los usuarios en el formulario de suscripción, antes de introducirlos en la base de datos.

Ejercicio 5

Para permitir que haya un usuario administrador y que mediante un formulario pueda dar de alta nuevos discos en la base de datos, inicialmente hay que comprobar que el usuario logeado es el administrador, y en ese caso lo redirijo a el formulario de insertar discos.
En **index.php** :

```
if($_SESSION['use']==='Admin') header("Location:insertDiskForm.php");
//echo "<a href='logout.php'> Logout</a> ";
```

En **procesarInsercionDisco.php** se procesa la insercción del disco en la base de datos.
La única dificultad que presenta este ejercicio es hacer que esos discos de la base de datos sean lo que se muestren después en la correspondiente sección. Para ello, dentro de una sección, por ejemplo, la clásica, obtengo todos los discos de esa sección en la base de datos con un método que he implementado, llamado **obtenerTodosDeSeccion(unaSeccion)**

```
public static function obtenerTodosDeSeccion($seccion)
{
 $conexion = self::conectar();
 try {
  $sql_query = $conexion->prepare("SELECT * FROM Discos WHERE Genero='$seccion'");
  $sql_query->bindParam(':Genero', $seccion);
  $sql_query->execute();


  $resultado=$sql_query->fetchAll(PDO::FETCH_ASSOC);
  return $resultado;
 }catch (PDOException $e) {
  echo "Error: " . $e->getMessage();
 }
 self::desconectar($conexion);
 }
```

y una vez obtenidos se van consultando sus campos y se muestran.

Ejercicio 6

Para mostrar las ventanas emergentes con las canciones de los discos cuando el ratón pasa por encima de estos, inicialmente pensé en una solución usando javascript y **alert** junto con **onMousseOver**, pero los alert no se pueden posicionar a voluntad, así que finalmente opté por una solución en CSS basada en **hover** y la propiedad display, que consiste en mantener un elemento hijo (un span) oculto, que será la lista de canciones, hasta que el ratón se posicione sobre el padre, que es la imágen del disco.
Para ello he creado una base de datos de canciones, y cuando el ratón se posiciona encima de la imágen, llamo a una función **getTracks(disco)** que me devuelve todas las canciones de un disco dado, si las hay.
La función completa, es:

```php
function getTracks($nomdisco){

        require_once('configuracion.php');
        try {
            $conexion = new PDO( DB_DSN, DB_USUARIO, DB_CONTRASENIA );
                    // set the PDO error mode to exception
            $conexion->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            $sql_query = $conexion->prepare("SELECT * FROM Canciones WHERE Nombre_Disco='$nomdisco'");
            $sql_query->bindParam(':Nombre_Disco', $nomdisco);
            $sql_query->execute();

            $resultado=$sql_query->fetch(PDO::FETCH_ASSOC);
            if (count($resultado)>1){
                $i = 0;
                foreach ($resultado as $track) {
                    if (!($i == 0)) {
                        echo "$i . $track " . "<br/>";
                    }
                    $i++;
                }
            }else echo "no hay canciones para este disco";
        }
        catch(PDOException $e)
        {
            echo "Error: " . $e->getMessage();
        }
        $conn = null;
}
    ?>
```

Aspectos Innovadores

Algunos aspectos novedosos para mí que han concurrido en esta práctica (aparte de la práctica en sí), han ido desde como sanitizar y validar usando PHP Filters, validar con javascript usando el document para obtener los valores de los elementos, las sesiones en PHP o pasar parámetros en las url hasta la propiedad hover y su uso en CSS o un repaso exprés de expresiones regulares.

Referencias

http://www.wickham43.net/hoverpopups.php http://jsfiddle.net/cor6bay6/1/ http://stackoverflow.com/questions/5210033/using-only-css-show-div-on-hover-over-a http://www.w3schools.com/php/php_sessions.asp http://www.w3schools.com/php/php_filter.asp http://emailregex.com/ http://www.tutorialspoint.com/javascript/javascript_form_validations.htm