Programación Web (2018-2019)
Doble Grado en Ingeniería Informática y Matemáticas
Universidad de Granada

Adaptación de la web del recomendador de libros.

Simón López Vico

3 de junio de 2019

LIBOOKS

Introducción.

En esta documento trataremos el desarrollo de los aspectos dinámicos sobre la página web *Libooks*. Comenzaremos hablando de los distintos aspectos principales a desarrollar pedidos en el PDF de la práctica, y después trataremos las características no pedidas introducidas en la web.

Antes de empezar, hablemos de las clases desarrolladas para realizar las consultas SQL. Dispondremos de una clase `DataObject`, la cual será la encargada de conectarse y desconectarse a la base de datos, así como devolver el valor de sus distintos atributos. Por cada tabla de la base de datos, se ha desarrollado una clase que hereda de `DataObject`. Esta clase será la encarga de realizar consultas sobre la base de datos así como insertar, modificar o eliminar elementos de la tabla. También incluirá algunas funciones necesarias para el desarrollo de la práctica. Todas estas clases se encuentran en `bookrecsysII/php_classes`

Aspectos principales:

1. Dotar de funcionalidad el formulario de registro en la web:

   Para empezar, creamos en la base da datos la tabla necesaria para guardar toda la información de un usuario:

```
create table usuarios(
  nick varchar(20) PRIMARY KEY,
  email varchar(50) UNIQUE NOT NULL,
  contrasenia varchar(20) NOT NULL,
  nombre_apellidos varchar(80),
  nacimiento date,
  pais varchar(30),
  libro_fav varchar(60),
  n_leidos int NOT NULL,
  imagen varchar(200)
);
```

   Tras esto, en `altausuario.php` cambiamos el *action* del formulario para especificar a qué documento se enviarán los valores introducidos, en nuestro caso a `registrar_usuario.php`. En este archivo, crearemos un objeto de la clase *Usuario* y llamaremos a la función `insertarUsuario(...)` con los valores introducidos en el formulario, los cuáles obtendremos con `$_POST['nombreDelInput']`. Además, dentro esta función se comprobará si alguno de los datos no obligatorios se ha dejado en blanco, y así dejar ese campo en la tabla a *NULL*. Finalmente, hacemos un *header* a la página principal, y el usuario ya podrá entrar en su cuenta personal.

2. Modificar la página principal para permitir identificarse al usuario y abrir así una sesión:

   Cambiamos el *action* el formulario creado en la primera práctica para acceder a la web, llamando así a `acceder.php`. Aquí se creará un objeto de la clase *Usuario* y se llamará a la función `validarUsuario(...)` con el nick y la contraseña introducidos en en el formulario de acceso; si la función devuelve `TRUE`, se creará una nueva sesión, y si devuelve otra cosa, se redireccionará a la página principal enviando información de este fallo.

```
$nick=$_POST['nick'];
$contrasenia=$_POST['contrasenia'];
$validacion=$usuario->validarUsuario($nick, $contrasenia);

if($validacion===TRUE){
  session_start();
  $_SESSION['conectado']=TRUE;
  $_SESSION['nick']=$nick;
  $_SESSION['n_leidos']=$usuario->get_n_leidos($nick);

  header("Location: index.php");
}else{
  header("Location: index.php?".$validacion);
}
```

   La variable `$validacion` podrá almacenar `TRUE`, `fail=1` y `fail=2`, utilizando estos dos últimos para mostrar mensajes de error si el usuario existe pero has fallado en la contraseña o si el usuario no existe, respectivamente.

   Para cerrar la sesión simplemente habrá que pinchar en el botón *Desconectar*, llamando así a `desconectar.php` que destruirá la sesión y eliminará los valores del array `$_SESSION[]`.

3. Mostrar cajita de iniciar sesión o desconectar:

   En todas las páginas de la web, comenzaremos comprobando si se ha iniciado sesión, para así comprobar qué mostrar en la esquina superior derecha. Definiremos una función `cajita($i)`, que dependiendo del valor que le pasemos (0 o 1) mostrará la cajita de iniciar sesión o de usuario loggeado, junto a su nombre y el número de libros leídos (realmente es el valor del número de libros introducidos u opinados). Por ejemplo, en `index.php`, definiremos la cajita en la cabecera de la siguiente manera:

```
<?php
  require_once("cajita.php");
```

```
  session_start();
  session_regenerate_id();

  if(!isset($_SESSION['conectado']))
    cajita(0);
  else if(isset($_SESSION['conectado'])
          && $_SESSION['conectado'] === TRUE)
    cajita(1);
?>
```

Hay ciertas páginas donde no puedes acceder si no estás conectado, por lo que en vez de llamar a `cajita(0)`, redireccionamos a `index.php` mostrando un mensaje para que el usuario acceda a la web.

4. Permitir crear de libros y valorarlos, así como modificar esta valoración:

   Para realizar esto, crearemos dos nuevas tablas en la base de datos, así como dos nuevas clases que hereden de `DataObject`:

   - La tabla *libros* almacenará en la base de datos el ISBN (clave principal), el título, el autor, la editorial, el año de publicación, el número de páginas, la descripción, el nick de quien haya introducido el libro y una imagen de portada. El ISBN se definirá con la sentencia `AUTO_INCREMENT`, para que cada vez que se introduzca un nuevo libro se le asigne un ISBN igual al último ISBN introducido más uno.

     En la clase *Libro*, definiremos funciones para introducir, modificar y eliminar libros así como para obtener todos los libros introducidos por cualquier usuario o un usuario en concreto y métodos para obtener el mayor y el menor ISBN (necesario para navegar entre todos los libros introducidos).

   - La tabla *opiniones* tendrá como clave principal el ISBN del libro junto a quién escribe la opinión, y almacenará los valores `opinion` y `n_estrellas`.

     En la clase *Opinion*, junto a las funciones de insertar y modificar, habrá métodos para obtener todas las opiniones de un libro o la opinión de un usuario en especial. Además, se definirá una función para borrar todas las opiniones de un libro, la cuál será necesaria para borrar un libro por completo.

   Para añadir un libro, simplemente habrá que pinchar en *Dar de alta nuevo libro*, que nos llevará a `altalibro.php` donde un formulario recogerá todos los valores necesarios para registrar un libro y como *action* los enviará a `registrar_libro.php`. En esta página se crearán tres objetos, *libro*, *opinion* y *usuario*. Con el objeto *libro* insertaremos un nuevo libro con los valores introducidos en el formulario y guardaremos el valor del ISBN de este libro con la función `getISBNlastInserted()`.

Con este ISBN y el valor de `$_SESSION['nick']`, añadiremos una nueva opinión y el número de estrellas dadas al libro. Tras esto, llamaremos a la función `nuevoLibroLeido(...)` definida en la clase *Usuario*, que sumará 1 al valor de `n_leidos`, y redireccionaremos a la página del libro recién añadido (`libroleido.php?v=ISBN`).

Para modificar la valoración, pincharemos en *Modificar libro* (si estamos en `libroleido.php`) o en valorar libro (si estamos en `libro.php`), que nos llevará a `valoracionlibro.php?v=ISBN`. En esta página se mostrará la opinión y valoración del usuario si ya se ha valorado el libro, o estará la opinión en blanco y el número de estrellas a 0 si no se ha valorado todavía. El *action* del formulario enviará información en la URL para determinar el ISBN del libro (`v=ISBN`) y avisar de si anteriormente este libro estaba valorado o no (`op=0/1`).

Este formulario enviará a `registrar_modificar_valoracion.php`, que realizará lo siguiente:

```
$isbn=$_GET["v"];

if($_GET["op"]==0){ // No estaba valorado
  $opinion->insertarOpinion(...);
  $usuario->nuevoLibroLeido($_SESSION['nick']);
  $_SESSION['n_leidos']=
      $usuario->get_n_leidos($_SESSION['nick']);
}else if ($_GET["op"]==1){ // Ya estaba valorado
  $opinion->modificarOpinion(...);
}

header("Location: libro.php?v=".$isbn);
```

Si el libro no estaba valorado, se añade la nueva opinión y se suma 1 al número de libros leídos; en cambio, si ya se había valorado, simplemente se modifica la opinión. Tras esto, se muestra el libro con la información ya modificada.

5. Que cualquier usuario pueda ver los libros que hay en el sistema dados de alta y expresar su opinión y poder modificarla en cualquier momento.

   En la páginas `mis_libros.php` mostraremos los libros que ha introducido el usuario que está loggeado actualmente en la columna de la izquierda, y todos los libros introducidos en *Libooks* por orden de antigüedad (el más antiguo en el final) en la columna de la derecha, junto a un botón para añadir un nuevo libro.

   Para mostrar la columna de la izquierda, llamamos a la función `getLibrosFrom(...)` pasándolo el nick de la sesión actual, que nos devolverá todos los libros que haya introducido este usuario. Tras ello, recorremos el array de libros obtenido con `foreach`, pintando cada uno de ellos en la columna de la izquierda, y definiendo un

hiperenlace a `libroleido.php?v=ISBN` para cada uno de ellos. Si no se devuelve ningún libro, pintamos un mensaje avisando de que no hay libros introducidos que nos lleve a `altalibro.php`.

La columna de la derecha la mostraremos de igual manera pero usando la función `getAllLibros('ISBN')`, donde el parámetro pasado a la función indica que el resultado se devuelva por orden descendente del valor del ISBN.

Si pinchamos en un libro de la columna de la izquierda, podremos modificar su opinión con el botón *Modificar Opinión* en todo lo bajo de la página (explicado en el punto anterior). Por otro lado, si pinchamos en un libro de la derecha, se nos mostrará el nombre de quién lo ha introducido junto a su imagen de usuario, y todos los comentarios introducidos sobre dicho libro, resaltando el comentario (si lo hubiese) del usuario loggeado actualmente.

Finalmente, para pasar de un libro a otro con los botones "Anterior" y "Siguiente", sumaremos o restaremos 1 al ISBN actual, y si dicho valor no existe como ISBN en la base de datos, volvemos a sumar o restar 1, así hasta que tengamos un ISBN válido (notar que los ISBN se van introduciendo con valores contiguos). Cuando tengamos un valor válido para el libro anterior y otro válido para el siguiente, se determinará el hiperenlace que lleva al anterior o siguiente libro con `libro.php?v=ISBN(previous/next)`. Si estamos en el último libro, al pulsar siguiente se mostrará el primero, y si estamos en el primero al pulsar anterior se mostrará el último.

6. Que el usuario activo pueda modificar sus datos personales.

   Si el usuario pincha en *Editar perfil* en la esquina superior derecha, se cargará la página `datospersonales.php` donde se mostrará la imagen del usuario y su nombre junto a todos los datos personales que haya introducido.

   Se mostrará un formulario ya rellenado con los valores actuales del usuario conectado; simplemente habrá que especificar en el *input* el `value=''<?php echo $valoractual?>''`. Para mostrar el valor actual del país será un poco más complicado, y para ello usaremos el siguiente script:

```
<script>
  <?php
    echo "var valor = '{$pais}';";
  ?>
  var indice=document.getElementById(valor).index;
  document.getElementById("pais").selectedIndex = indice;
</script>
```

   Para modificar los datos, se llamará a `modificar_usuario.php` al pinchar sobre *Modificar*, donde se usarán los métodos `cambiarContrasenia(...)` y `modificarDatos(...)`.

Tras ello, se redireccionará a `datospersonales.php?modified` y se mostrará un *alert* avisando de que los datos han sido modificados.

7. Validar la corrección de todos los formularios.

   En `altausuario.php` definimos la función `validateForm()`, la cuál llamaremos en el formulario con `onsubmit=‘‘return validateForm()’’`. Esta función mostrará un *alert* y devolverá `false` cada vez que se incumpla una restricción en los campos introducidos del formulario (que se haya dejado un campo obligatorio en blanco, que se haya superado el límite de caracteres, que las dos contraseñas introducidas no sean la misma...). De la misma manera, se harán estas comprobaciones en `datospersonales.php`, `altalibro.php` y `valoracion_libro.php`.

   Hay dos cosas a comprobar más “complicadas”: que el email introducido sea un email válido y que el valor de un campo sea numérico. Para comprobar la validez de un email definiremos en JavaScript la función `validateEmail(email)`, que comprobará mediante una expresión regular si el email es del tipo `[algo]@[algo].[algo]`. Por otra parte, para comprobar que hemos introducido un valor numérico usaremos la función `parseInt(valor)` y si devuelve `NaN` es porque el valor no era numérico.

```
<script>
  function validateForm(){
    [...]
    var anio=document.forms["altalibro"]["anio"].value;
    anio_int=parseInt(anio);
    if(isNaN(anio_int) && anio!=""){
      alert("No has introducido un anyo valido.");
      return false;
    }
    [...]
  }
</script>
```

8. Que el usuario pueda subir su foto a la web cuando se dé de alta y al pasar el ratón por encima, aparecerá una ventana emergente con los títulos de los libros que ha subido.

   Ya que no teníamos permisos en el servidor para subir imágenes vía PHP, se han introducido 6 imágenes por defecto para que el usuario pueda elegir una de ellas como foto de perfil. Pondremos estas imágenes en el formulario de alta de usuario con botones de tipo `<input type=‘‘radio’’...>` y en la base de datos guardaremos el nombre de la imagen (`user1.png`, `user2.png`...). De la misma manera se le asignará una portada a un libro cuando se de de alta. Estas imágenes no se podrán modificar.

Para implementar la ventana emergente que aparece y desaparece al pasar el ratón, hemos definido un `section` al cuál le hemos asignado en el documento CSS `position:absolute;`, por lo que se mostrará sobre el resto de elementos. Ajustaremos sus valores de posición para que se encuentre sobre la foto y pondremos `display:none;` para que por defecto no se muestre.

Dentro de este `section` llamaremos con PHP al método de la clase *Libro* para obtener los libros de un usuario y con `foreach` mostraremos el nombre y autor de cada uno de ellos. Si no hay ningún libro introducido por dicho usuario, se mostrará un mensaje para avisar sobre ello. Al final de esta ventana emergente se mostrará el número de libros que ha introducido/valorado el usuario.

Para hacer que esta ventana se muestre al pasar el ratón, definiremos las siguientes funciones con JavaScript:

```
<script>
  function MouseOver(obj) {
    document.getElementById("libros_subidos").style.display
                                                  = 'block';
  }

  function MouseOut(obj) {
    document.getElementById("libros_subidos").style.display
                                                  = 'none';
  }
</script>
```

Para que estas funciones actúen al pasar el ratón sobre la imagen de usuario, definiremos la imagen con `onmouseover=‘‘MouseOver(this)’’` y `onmouseout=‘‘MouseOut(this)’’`. Esta funcionalidad también se ha definido en `libro.php` pasando el ratón sobre la imagen del usuario que ha introducido el libro.

Elementos adicionales y restricciones:

- Cada vez que el usuario introduce un nuevo libro o valora un libro que no había valorado anteriormente, se le suma 1 al contador de libros leídos (se supone que si lo valora es porque lo ha leído...). Si modifica su opinión sobre un libro (aunque se utilice la misma página que para valorarlo por primera vez) NO se suma 1 al número de libros leídos.
- El usuario puede eliminar un libro que él haya introducido. Tras pulsar el botón de eliminar aparecerá una ventana para confirmar la acción (añadido con JavaScript).

- Cada vez que se elimina un libro o se modifican los datos de usuario, se muestra un *alert* avisando de ello.
- Se calcula la valoración media real para cada libro; se muestra un redondeo al entero más próximo en número de estrellas, y a su lado el valor real entre paréntesis (con dos decimales).
- Como se ha comentado anteriormente, podremos ver los libros que ha introducido un usuario que aparezca en `libro.php`.
- La opinión que haya dado el usuario actual sobre un libro se resalta sobre el resto de opiniones que aparezcan.
- Aunque los ISBN de todos los libros se introduzcan de manera contigua, si se elimina un libro que se encuentre entre medias dejarán de ser contiguos. Esto se tiene en cuenta a la hora de asignar los botones *Siguiente* y *Anterior* en la página `libro.php`
- La web mantiene un *responsive-design* tras la introducción los nuevos elementos.

Restricciones:

Se han establecido distintas restricciones para que un usuario no pueda acceder a ciertos sitios de la web que no debiera acceder. Para ello, utilizaremos la funcionalidad de enviar variables mediante la URL, que comprobaremos para ver los tipos de errores. Todos los errores se mostrarán con un *alert* tras cargar la web con el siguiente código JavaScript:

```
<script type='text/javascript'>
  window.onload = function(){alert('[Mensaje de error]');}
</script>
```

Las restricciones definidas serán las siguientes:

- Si un usuario intenta acceder a cualquier sitio distinto a `index.php` y `altausuario.php` sin haberse identificado, se le redireccionará a `index.php` mostrándosele un mensaje de que necesita *logearse* para acceder ahí.
- Si un usuario conectado intenta acceder a `altausuario.php`, se le redireccionará a `datospersonales.php`.
- Si se intenta acceder a `libro.php`, `libroleido.php` o `valoracion_libro.php` mediante un ISBN que no existe, se redireccionará a `mis_libros.php` y se avisará de que dicho libro no existe.
- Si se intenta acceder a `libroleido.php` con el ISBN de un libro que no ha introducido el usuario actualmente *logeado*, se redireccionará a `mis_libros.php` avisando de que no se es dueño de dicho libro.