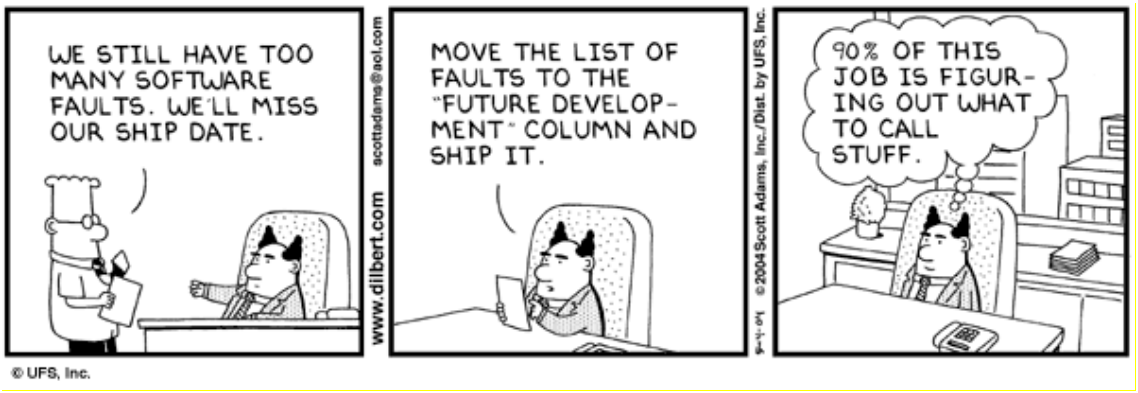
Comercio Electrónico

Práctica 6: Webs dinámicas



José Luis Salazar
jsalazar@unizar.es

Antonio Sanz
ansanz@unizar.es

Rafael del Hoyo
rdelhoyo@ita.es

Objetivo de la Práctica

En esta práctica se mostrarán varios métodos de caonsguir dinamismo en página web. Para ello comenzaremos por estudiar un lenguaje de programación orientado a web, que permita el establecer todas las funcionalidades deseadas en el proyecto de comercio electrónico de forma rápida y eficiente. Finalmente utilizaremos servicios web que incrustados en nuestra página no den información dinámica provenientes de servidores ajenos

¿Qué hay preparar de forma previa a la práctica?

Se van a emplear la totalidad de los servicios desplegados en la Práctica 3, por lo que será necesario verificar que tanto el servidor web como la base de datos y el servidor de FTP están funcionando correctamente. Dado que se trabajará con las tablas definidas en la práctica 3, será fundamental el tener su estructura correctamente definida y comprendidos todos sus campos.

También será recomendable, dado que se va a trabajar con código HTML, el tener frescos los conceptos de HTML vistos en la práctica anterior.

¿Cuál es el resultado de la práctica?

Se obtienen como resultado dos ejemplos de formularios web programados con HTML y PHP, siendo estos formularios capaces de interaccionar con la base de datos. También sabremos incrustar servicios ajenos en nuestra página web.

¿Qué se aprende con esta práctica?

Se aprenden a realizar mediante HTML formularios web que permitan a los usuarios introducir datos en nuestros repositorios de información, así como los rudimentos del lenguaje PHP que nos permiten conectarnos a bases de datos y generar código HTML de forma dinámica.

Introducción

Después de haber aprendido en la práctica anterior los principios del lenguaje HTML para poder mostrar información a nuestros usuarios, vamos a repasar las herramientas de las que disponemos para poder interactuar con el usuario en sus dos aspectos principales: Poder recibir información del mismo, y poder mostrarle la información que desea.

La recepción de información se realiza mediante formularios web, y la información que desea ver el usuario se recuperará de la base de datos y se generará “al vuelo” una página web con dicha información. En este caso lo interesante es disponer de un programa que sea capaz de construir una página en HTML que sea diferente para cada usuario, evitando tener que tener todos los resultados posibles en nuestro servidor.

Formularios web

Es posible el introducir código dentro de una página HTML que muestre una serie de cajas en los que los usuarios puedan introducir o seleccionar información. A este tipo de código se le denomina “formulario web”, y son fundamentales para nosotros porque nos permiten interactuar con el usuario. Podemos ver un ejemplo de un formulario realmente sencillo en el código siguiente:

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>
      Introduzca su nombre:
<FORM ACTION="muestra.php" METHOD="POST">
 <INPUT TYPE="text" NAME="nombre"><BR>
<INPUT TYPE="submit" VALUE="Enviar">
</FORM>
</body>
</html>
```

Esta web muestra una caja de texto y un botón de enviar para que el usuario introduzca su nombre y envíe los datos. La etiqueta HTML <FORM> es la que define el comienzo y el fin del formulario, siendo necesario que tenga al menos dos valores:

- ACTION : Nombre del programa al que se enviarán los datos recogidos. En este caso cuando se envíen los datos se ejecutará el programa *muestra.php* (los formularios no distinguen entre PHP, ASP o cualquier otro lenguaje de programación).

- METHOD: Método por el cual se envían (GET o POST). GET envía los datos en la misma URL, mientras que POST lo hace mediante envío directo. Por motivos de seguridad es más recomendable emplear POST (nuestros datos no quedan en la URL de los navegadores).

Los diferentes campos que puede tener un formulario (que veremos a continuación) vienen precedidos por la etiqueta <INPUT TYPE>, siendo fundamental el *<INPUT TYPE="submit" VALUE="Enviar">* ya que es el que contiene el botón de “Enviar”.

Todos los campos que tiene un formulario están identificados por el valor NAME, que indica el nombre de la variable que se pasará a nuestro programa en PHP. Es por ello altísimamente recomendable el elegir con cuidado esos nombres para que se ajusten perfectamente a los que usaremos en nuestro código PHP.

Cajas de texto

```
<input type="text" size="40" maxlength="64" NAME="nombre" />
```

Las cajas de texto nos permiten recoger texto de los usuarios, y son vitales para que puedan por ejemplo introducir sus datos personales para finalizar una compra.

Es interesante el hecho de que podemos limitar el tamaño máximo del campo de texto (que por motivos de seguridad siempre es recomendable).

Cajas de texto multilínea

```
<textarea rows="4" wrap="ON" maxlength="255"
name="comentario"></textarea>
```

En el caso de que queramos dejar un cierto espacio a los usuario (por ejemplo, para que escriban un comentario o una reclamación), podemos emplear una caja de texto multilínea. El valor *rows* indica el número de filas iniciales que tiene el texto, el valor *wrap* si se permite o no scroll lateral, y el *maxlength* su tamaño máximo.

Cajas de texto de contraseña

```
<input type="password" name="password" size="20" />
```

Este campo es imprescindible cuando queremos que los usuarios introduzcan su nombre de usuario y contraseña para autentificarse en nuestro sistema. El funcionamiento es igual que el de un campo de texto, solo que no se muestra el texto tecleado.

Botones Radio

```
<input type="radio" name="informacion" value="Si" />
<input type="radio" name="informacion" value="No" checked="checked" />
```

Estos botones son muy útiles cuando queremos que el usuario elija una sola opción entre varias disponibles. El valor *value* indica lo que se le pasará al PHP, y podemos preseleccionar un valor dado con el valor *checked*.

Checkboxes

```
<input type="checkbox" value="Extra 1" checked="checked"
name="opciones_extra" />
<input type="checkbox" value="Extra 2"  name="opciones_extra" />
<input type="checkbox" value="Extra 3"  name="opciones_extra" />
<input type="checkbox" value="Extra 4"  name="opciones_extra" />
```

Las checkboxes sirven perfectamente para dar al usuario a elegir cero o varios elementos de una lista (es importante que todos los valores tengan el mismo nombre de variable como los botones radio, ya que de lo contrario serán interpretados como elementos independientes del formulario).

Listas desplegables

```
<select name="regalo">
<option value="1">Regalo 1</option>
<option value="2">Regalo 2</option>
<option value="3">Regalo 3</option>
</select>
```

Las listas desplegables son una forma muy cómoda de dar al usuario la opción de que elija entre uno solo de varios valores (y de evitarle que tenga que escribirlo en una caja de texto y tener que corregir lo que haya escrito mal).

Botones de Enviar y Borrar

```
<input type="submit" value="Enviar" />
<input type="reset" value="Borrar" />
```

El botón de *submit* recoge todos los datos y los envía al script indicado en el formulario, mientras que el de *reset* borra todos los datos para que el usuario pueda introducirlos de nuevo. Es posible definir nuevos botones para que con código JavaScript, por ejemplo, podamos realizar acciones sobre el propio formulario.

Ejemplo de formulario completo

```
<html>
<head>
  <title>Ejemplo de Formulario completo</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>

<FORM ACTION="muestra.php" METHOD="POST">
Introduzca su nombre:
<br><input type="text" size="40" maxlength="64" NAME="nombre" />
<br><br>
Por favor, introduzca su contraseña<br>
<input type="password" name="password" size="20" />
<br><br>
Deje aqui su comentario sobre la tienda<br>
<textarea rows="4" wrap="ON" maxlength="255"
name="comentario"></textarea>
<br><br>
¿Desea informacion sobre nuestros productos?<br>
Sí <input type="radio" name="informacion" value="Si" />
No <input type="radio" name="informacion" value="No" checked="checked" />
<br><br>

Por favor, marque las opciones extra que desea en nuestro producto
<br>Opcion 1<input type="checkbox" value="Extra 1" checked="checked"
name="opciones_extra" />
<br>Opcion 2<input type="checkbox" value="Extra 2"  name="opciones_extra" />
<br>Opcion 3<input type="checkbox" value="Extra 3"  name="opciones_extra" />
<br>Opcion 4<input type="checkbox" value="Extra 4"  name="opciones_extra" />
<br><br>

Elija su regalo promocional ! <br>
<select name="regalo">
<option value="1">Regalo 1</option>
<option value="2">Regalo 2</option>
<option value="3">Regalo 3</option>
</select>
<br><br>

<input type="submit" value="Enviar" />
<input type="reset" value="Borrar" />

</FORM>
</body>
</html>
```

PHP

PHP es un lenguaje principalmente orientado a web, muy empleado en la actualidad para el desarrollo de aplicaciones web de tamaño pequeño y mediano (aunque también se está empezando a emplear en grandes aplicaciones gracias al soporte para clases y objetos). Las principales características de PHP son:

- Sencillez: La sintaxis básica es muy sencilla de aprender y manejar.
- Generación de código HTML dinámica: Veremos que con PHP es trivial el generar unos resultados en HTML.
- Conexión a bases de datos integrada: El propio PHP tiene funciones integradas para la conexión a bases de datos estándar como MySQL o PostgreSQL.

Como programas gratuitos para desarrollar páginas en PHP podemos tomar el PHP Designer en entornos Windows o el Eclipse PHPIDE en entornos Linux o Windows (Quanta también soporta PHP pero solo está en entornos Linux).

Un programa en PHP puede (y suele casi siempre) el tener adjunto código HTML. Es una buena costumbre el intentar separar lo más posible el HTML del PHP en un programa, porque se facilita su legibilidad y su futura adaptación. Un ejemplo totalmente básico del programa que procesa los datos enviados por el formulario anterior podría ser éste:

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>
El nombre que ha introducido es:
<?php echo $HTTP_POST_VARS ['nombre'] ; ?>
<br>
</body>
</html>
```

El hecho de que esa página tenga ese texto en negrita la convierte en una página dinámica PHP (una de las más sencillas que podemos generar). Analicemos dicha página:

- El inicio y fin de un código PHP se delimita mediante las etiquetas <?php y ?> respectivamente. Es vital distinguir en una página PHP cuando estamos en código PHP y cuándo estamos en código HTML (porque se representan de forma diferente).
- $HTTP_POST_VARS[‘nombre’] → Cuando en un formulario enviamos información a través del método POST, el servidor recoge las variables y las almacena en el array $HTTP_POST_VARS. $_POST[‘nombre’] es igual al valor de la caja de texto “nombre” del formulario anterior. (Si hubiéramos empleado GET, sería $HTTP_GET_VARS[‘nombre’].

- Echo → Esta es la orden que nos va a permitir generar el código dinámico HTML siempre que lo deseemos. Todo lo que pongamos entre comillas después de la orden echo será sacado por pantalla como respuesta del script, pudiendo sin problema meter código HTML dentro de esa salida. Un ejemplo podría ser éste:

  *echo " Tu nombre es <b> $_POST['nombre']<b>
" ;*

  Una de las cosas a tener en cuenta cuando empleamos el comando echo es que las comillas marcan principio y final de la salida, por lo que si en nuestro código PHP insertado tenemos comillas (" ") , tendremos que cambiarlas por comillas simples (' ').

  Es también fundamental recordar que toda línea de código PHP debería de estar finalizada con el símbolo punto y coma (;) por imposición del lenguaje.

PHP Designer – Configuración para PHP

Para que el PHP Designer soporte correctamente PHP tenemos que indicarle dónde están situados tanto el ejecutable de PHP como el fichero de configuración php.ini.

Esta configuración puede realizarse desde *Options → Editor Preferentes → Debug* , y los ficheros de PHP están en *%XAMPP%/php.*

PHP – Principios básicos del lenguaje

- Etiquetas de inicio y fin del código PHP: <?php y ?>
- Carácter de final de línea PHP : Punto y coma (";")
- Variables de tipo cadena: $mi_nombre (ya sea una cadena, un entero o una palabra).

  $nombre = "Pepe" ; $nombre = "Hola caracola" ; $nombre= "123";

  (PHP detecta el tipo de variable y le asigna memoria automáticamente, por lo que no es necesario especificar el tipo de variable)
- Comentarios : Al estilo de C , o con " // " en cada línea:

  // Esto es un comentario en una línea
  /* Esto es otro comentario */
- Concatenar variables : $a . $b ;

  $nombre_completo = $apellido1 . " ". $pellido2 . " , " . $nombre ;
- Operaciones con variables : Sumar, restar, incrementar …

```
$c = $a + $b + 5 ;    $c++ ;    $c = $c * 2 ;
```

- Operaciones de comparación: Es igual, es distinto a … AND, OR

```
$c == “5” ,   $c!= $b ,   $a && $b,  $a || $b
```

(Muy importante !!!. En PHP la comparación de igualdad se realiza con DOS símbolos de “=”. Si ponemos uno solo, esa condición se cumplirá siempre).

- Estructura de control “IF” :

```
if ( $stock != 0) {
      echo “Tenemos productos”;
}
else {
      echo “Lo sentimos, no tenemos más productos”;
}
```

- Estructura de control “FOR” :

```
for ( $i = 1: $i <= $numero_productos ; $i++ ) {
      echo “Producto numero $i “;
}
```

- Estructura de control “WHILE” :

```
while ( $i < $numero_productos ) {
      echo “Producto número: $i “ ;
      $i++ ;
}
```

- Usando funciones y código PHP de otros ficheros :

```
require("header.php");
require(“funciones.php”):
```

Conexión a una base de datos (usando MySQL)

```
// Definimos los parámetros
$host = "localhost";
$user = "root";
$password = "";
$database = "cdcol";
// Nos conectamos a la BD
$conexion = mysql_connect($host, $user, $password);
mysql_select_db($database, $conexion);
// Preparamos la consulta (ojo a la concatenación con el . para dar claridad)
$consultaSQL ="select ";
$consultaSQL .="id";
$consultaSQL .=", jahr";
$consultaSQL .=", interpret";
$consultaSQL .=", titel ";
$consultaSQL .="FROM ";
$consultaSQL .=$_POST ['nombre'];
$consultaSQL .=" ORDER BY jahr;" ;
$resultado = mysql_query($consultaSQL, $conexion);
if (!$resultado) {
   die('Could not query:' . mysql_error()); }
// Obtenemos el número de resultados
$numFilas = mysql_num_rows($resultado);
// Recogemos los resultados. Estos se almacenan en una matriz de forma
// que  cada fila tiene los resultados de un producto, y cada columna los
// resultados de una categoría pedida.
$id = mysql_result($resultado, 0, 0);
$jahr = mysql_result($resultado, 0, 1);
$interpret = mysql_result($resultado, 0, 2);
$titel = mysql_result($resultado, 0, 3);
echo "$id ";
echo "$jahr ";
echo "$interpret ";
echo "$titel ";
// Cerramos la conexión y liberamos recursos
mysql_free_result($resultado);
mysql_close();
```

Google Maps

Google Maps (http://maps.google.com/) es un servicio web de mapas de todo el mundo que pueden ser vistos con cualquier navegador. Google Maps tiene como características particulares las siguientes :

- Los usuarios pueden crear mapas personalizados y compartirlos (por ejemplo, con los lugares turísticos más interesantes de una ciudad).
- Google Maps tiene una API (Application Programming Interface) que permite a desarrolladores externos el crear páginas web aprovechando la información de mapas que posee Google

Google Maps emplea de forma fundamental las siguientes tecnologías: XML, XHTML, AJAX y XMLHttpRequest, lo cual lo hace perfecto como ejemplo de servicio

web. Se puede encontrar más información acerca de la tecnología que hay por debajo de Google Maps aquí (http://blog.grimpoteuthis.org/2005/02/mapping-google.html).

Funcionamiento básico

Para ver todo el funcionamiento de Google Maps será necesario poseer una cuenta de Google (se puede usar cualquiera o generar una cuenta nueva accediendo a https://www.google.com/accounts/NewAccount).

El uso del interface es bastante intuitivo y sencillo, por lo que no debería de ser problemático el manejarlo (se anima al alumno a que lo experimente por sí mismo o que acceda a la ayuda en (http://maps.google.com/support/?hl=en).

En esta parte de la práctica cada alumno creará su propio mapa accediendo a "My Maps → Create new map". Es posible crear marcadores de posición con la herramienta con forma de gota azul, que tienen información adicional.

Se propone como trabajo que los alumnos creen un mapa público y coloquen media docena de marcadores con información relativa a su negocio electrónico (localización de sucursales, lugares de interés, etc …). Una vez creado, es posible el obtener la URL de este mapa en concreto accediendo a "Link to this page" (en la parte superior derecha) y poder colocarla en nuestra web.

Trabajo a realizar durante la práctica

Función de inserción de productos en el catálogo

Se tendrá que generar un formulario web llamado *nuevo_producto.html*, que contenga los campos necesarios para introducir un nuevo producto dentro de la tabla de productos del proyecto de comercio electrónico.

A continuación, se tendrá que crear un programa en PHP denominado *ins_prod.php* que recoja los datos del formulario y genere una consulta SQL que los introduzca en la BD.

Función de lectura del catálogo

Se tendrá que generar un formulario web llamado catalogo.html, que permita a un cliente ver los productos existentes en el catálogo ordenados al menos o por orden alfabético o por precio.

Este formulario llamara a muestra_catalogo.php, que realizará la consulta a la base de datos y mostrará los resultados mediante una tabla HTML generada de forma dinámica.

Función localización y meteorología de la región

Se tiene que generar una página web con un mapa creado en Google maps y con enlace ampliado a dicho sitio en el que se refleje una necesidad cubierta por dicho mapa (localización de la empresa, del sitio de almacenaje, tiendas reales, etc.). Además se añadirá la información meteorológica de una de las localidades que aparezcan en dicho mapa a través de un servicio web incrustado en la página y proveniente de www.aemet.es.

Enlaces y Bibliografía de interés

Tutoriales de PHP y Formularios Web
http://www.desarrolloweb.com/php/
http://www.webestilo.com/php/

Manual de PHP en español
http://es2.php.net/manual/es/index.php

"PHP and MySQL Web Development"
Luke Welling & Laura Thomson - Ed Sams Publishing – ISBN 0672326728

PHP Designer
http://www.mpsoftware.dk/phpdesigner.php

Eclipse PHPIDE
http://www.eclipse.org/php