Desarrollo web con PHP y MySQL

Ejemplos prácticos

Piero Berni Millet

PID_00155711

Universitat Oberta de Catalunya
www.uoc.edu

Índice

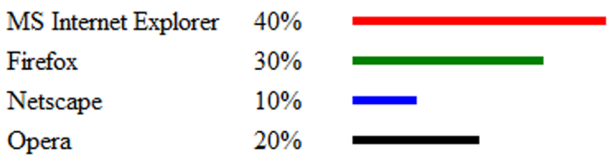
1. Gráfico de barras con PHP y HTML

Combinando PHP con HTML podemos generar fácilmente gráficas estadísticas de barras para datos porcentuales sin necesidad de recurrir a un *plug-in* extra de Flash o a una imagen GIF pregenerada dinámicamente con una librería específica de PHP.

Vamos a ver cómo se construye un gráfico de barras para una encuesta sobre navegadores de Internet, valiéndonos simplemente de las propiedades de las tablas HTML para emular dicho gráfico.

Navegadores	% de votos	Color barra gráfica
MS Internet Explorer	40	*red*
Firefox	30	*green*
Netscape	10	*blue*
Opera	20	*black*

¿Cual es tu navegador de Internet favorito?



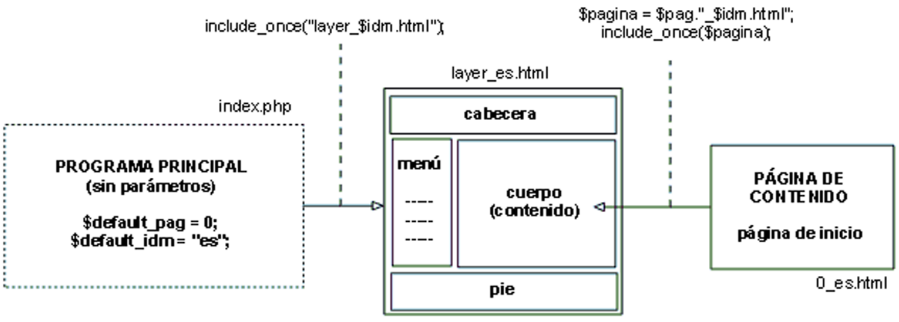
Programa: encuesta.php
Llamada: http://localhost/encuesta.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
       <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
       <title>Gráfico de barras dinámico con PHP y MYSQL</title>
</head>
<body>
<?php
// Array multidimensional con los datos de la encuesta y el color de la barra:
// Navegador, % votos, color barra gráfica
 $datos = array(
    array( "MS Internet Explorer", 40, "red" ),
    array( "Firefox", 30, "green" ),
    array( "Netscape", 10, "blue" ),
    array( "Opera", 20, "black" )
 );
?>
<h3>¿Cuál es tu navegador de Internet favorito?</h3>
<table width="600" cellspacing="0" cellpadding="2">
       <?php
        # Leo datos del array
        foreach( $datos as $d ) {
            $navegador = $d[0];
            $porcentual = $d[1];
            $color = $d[2];
       ?>
       <tr>
             <td width="25%"><?=$navegador?></td>
             <td width="10%"><?=$porcentual?>%</td>
             <td>
                   <!--Barras gráficas -->
                   <table width="<?=$porcentual?>%" bgcolor="<?=$color?>">
                         <tr><td> </td></tr>
                   </table>
             </td>
       <tr>

       <?php } ?>
</table>
</body>
</html>
```

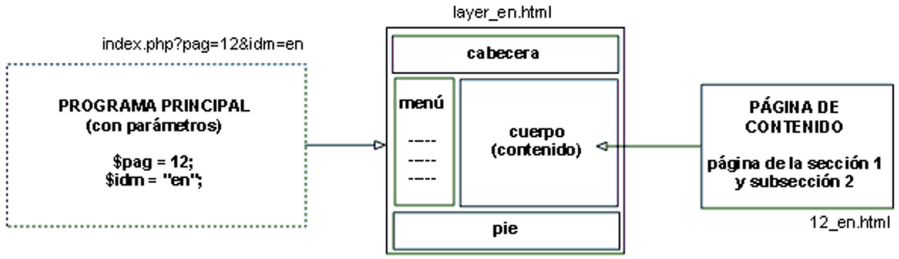
2. Web dinámica multilenguaje

Combinando PHP con HTML podemos crear sitios web dinámicos multilenguaje con un alto grado de productividad. El sitio web de este ejemplo está pensado para soportar tres o más idiomas: español (es), inglés (en) y catalán (cat). Los contenidos se encuentran desvinculados de la plantilla web que controla el diseño, la presentación y la navegación, lo que simplifica el mantenimiento del sitio cuando se modifican o se añaden nuevos temas.

Estructura de la web multilenguaje



Abro la página de inicio del sitio (home page)



Abro desde el menú una página de contenido para el idioma seleccionado

El programa principal (index.php)

Se accede al sitio mediante el *script* principal **index.php**. El recorrido por las diferentes páginas de un idioma determinado se controla con los parámetros 'pag' e 'idm' que se pasan en la URL.

Por ejemplo:

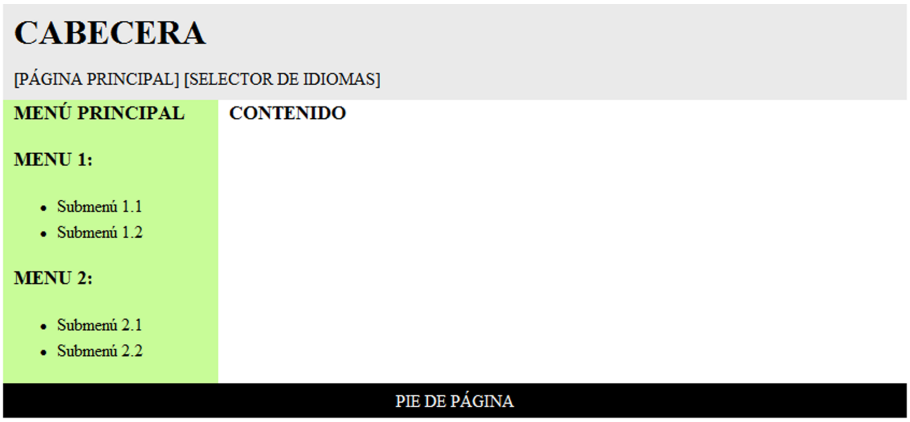
- http://host/index.php Si no se incluyen los parámetros en la URL se carga la página principal del sitio en el idioma predeterminado (ver index.php).

- http://host/index.php?pag=01&idm=es En este caso, los parámetros indicados cargarán la página de contenido 01 (01_es.html) para el idioma es (español).

El código del programa principal es el siguiente:

```
<?php
        // CARGO PARÀMETROS PASADO POR EL URL
        # Por ejemplo: http://host/index.php?pag=01&idm=es

        # Página del sitio web: pag=[número de la página a abrir]
        # Si no se ha especificado una página de contenido
        # cargo el contenido de la página inicial (0_$idm.html)
        $default_pag = 0;
        $pag = isset($_GET['pag']) && !empty($_GET['pag']) ? $_GET['pag'] : $default_pag;

        # Idioma del sitio web: idm=[es,en,cat]
        $idm = isset($_GET['idm']) && !empty($_GET['idm']) ? $_GET['idm'] : "";

        # Si no se ha pasado el idioma, o el indicado no está entre
        # los predeterminados, selecciono el idioma por defecto (es)
        $default_idm = "es";
        $idm = ($idm == "es" || $idm == "en" || $idm == "cat") ? $idm : $default_idm ;

        # Abro la plantilla para el idioma seleccionado
        include_once("layer_$idm.html");
?>
```

La plantilla (layer_$idm.html)



La plantilla del sitio se estructura en cuatro secciones:

- **Cabecera**. Espacio reservado para el título del sitio con su *banner*. En la parte inferior izquierda se colocan el enlace de "ir a la página de inicio", más los tres enlaces que permiten seleccionar un idioma.

```
<a href="index.php?pag=0&idm=es">Página de inicio</a>   
<a href="index.php?pag=0&idm=es">Español</a> |
<a href="index.php?pag=0&idm=en">Inglés</a> |
<a href="index.php?pag=0&idm=cat">Catalán</a> |
```

- **Menú principal**. La columna de la izquierda está reservada para el menú de navegación, que se estructura en dos niveles (sección y subsecciones). Haciendo clic en los enlaces del menú se abren las páginas de contenido en el espacio central de la plantilla.

```
<h3>MENU 1:</h3>
<ul>
        <li><a href="index.php?pag=11&idm=es">Submenú 1.1</a></li>
        <li><a href="index.php?pag=12&idm=es">Submenú 1.2</a></li>
</ul>
<h3>MENU 2:</h3>
<ul>
        <li><a href="index.php?pag=21&idm=es">Submenú 2.1</a></li>
        <li><a href="index.php?pag=21&idm=es">Submenú 2.2</a></li>
</ul>
```

- **Contenido**. Aquí se incrustan dinámicamente los contenidos del sito. Por ejemplo, los ficheros del sitio en español tienen la siguiente nomenclatura: 0_es.html (página de inicio o de bienvenida), 21_es.html (la página de contenido del la sección 2.1 del menú 2.

- **Pie de página**. Espacio reservado para los datos de contacto del sitio (dirección, correo electrónico).

Puesto que el sitio web soporta tres idiomas, la plantilla deberá triplicarse para cada uno de ellos con las nomenclaturas: layer_es.html, layer_en.html, y layer_cat.html

Incrustar los contenidos en el cuerpo de la plantilla

El mecanismo que controla las páginas de contenido y las incrusta en la sección Contenido se basa un pequeño *script* de PHP que debe residir en esa posición de la plantilla del sito. Este programa comprueba si existe la página de contenido en el sistema de archivos del servidor web. En caso de no existir, por ejemplo, el contenido 2.1_es.html, se muestra un mensaje de error.

```
<p><b>CONTENIDO</b></p>
<?php

# Ruta absoluta al directorio del sitio en el sistema de ficheros
$path = "/var/www/web/";
# Nombre del fichero de contenido
$pagina = $pag."_$idm.html";

# TEST: ¿existe fichero de contenido?
$file = $path.$pagina;
if(file_exists($file))
     include_once($pagina);
    else
     echo "El archivo de contenido $pagina no existe";
?>
</div>
```

El código completo de la plantilla HTML es el siguiente:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
   <title>Web dinàmica multillenguatge</title>
   <style type="text/css">


      body{
         margin:0;
         padding:0;
         line-height: 1.5em;
      }


      b{font-size: 110%;}
      em{color: red;}


      #maincontainer{
         width: 840px; /*Width of main container*/
         margin: 0 auto; /*Center container on page*/
      }


      #topsection{
         background: #EAEAEA;
         height: 90px; /*Height of top section*/
      }


      #topsection h1{
         margin: 0;
         padding-top: 15px;
      }


      #contentwrapper{
         float: left;
         width: 100%;
      }
```

```
      #contentcolumn{
         margin-left: 200px; /*Set left margin to LeftColumnWidth*/
      }

      #leftcolumn{
         float: left;
         width: 200px; /*Width of left column*/
         margin-left: -840px;
         /*Set left margin to -(MainContainerWidth)*/
         background: #C8FC98;
      }

      #footer{
         clear: left;
         width: 100%;
         background: black;
         color: #FFF;
         text-align: center;
         padding: 4px 0;
      }

      #footer a{
         color: #FFFF80;
      }

      .innertube{
         margin: 10px; /*Margins for inner DIV inside each column
         (to provide padding)*/
         margin-top: 0;
      }

   </style>

</head>

<body>
   <div id="maincontainer">

      <div id="topsection">
         <div class="innertube"><h1>CABECERA</h1>
            <p>[PÁGINA PRINCIPAL] [SELECTOR DE IDIOMAS]</p>
         </div>
      </div>

      <div id="contentwrapper">
         <div id="contentcolumn">
            <div class="innertube">
               <p><b>CONTENIDO</b></p>
            </div>
         </div>
      </div>
```

```
      <div id="leftcolumn">
         <div class="innertube">
            <p><b>MENÚ PRINCIPAL</b></p>
            <h3>MENU 1:</h3>
            <ul>
               <li>Submenú 1.1</li>
               <li>Submenú 1.2</li>
            </ul>
            <h3>MENU 2:</h3>
            <ul>
               <li>Submenú 2.1</li>
               <li>Submenú 2.2</li>
            </ul>

         </div>
      </div>


      <div id="footer">
         PIE DE PÀGINA

      </div>
   </div>
</body>
</html>
```

3. Formulario para enviar los datos a una cuenta Gmail

Generalmente, cuando se rellenan los campos de un formulario HTML y se envían al servidor mediante un *script* de PHP, los datos se registran en una base de datos. En otros casos, lo que se suele hacer es, tras procesarlos desde el lado del servidor, enviarlos a una cuenta de correo electrónico. Típicamente para el envío de correo con PHP se utiliza la función mail(), pero esta función tiene varias limitaciones, por ejemplo, hay que tener una cuenta de usuario registrada en el servidor que aloja la web y la función mail() no soporta el envío de adjuntos.

Afortunadamente, los datos recogidos mediante un formulario HTML pueden ser enviados a su destinatario mediante un servicio de correo externo tipo Gmail. Para ello, es necesario utilizar una clase de PHP llamada PHPMailer que facilita el envío de correo, añadiendo funcionalidad en el envío de correos con adjuntos, en formato HTML y con diferentes codificaciones, soporte para imágenes embebidas, *headers* personalizados y además funciona con múltiples servidores de correo, entre los que se encuentra Gmail.

Para llevar a cabo este ejemplo práctico se necesita tener una cuenta de correo Gmail y bajarse la clase phpmailer para php5/6 de la página de sus desarrolladores.

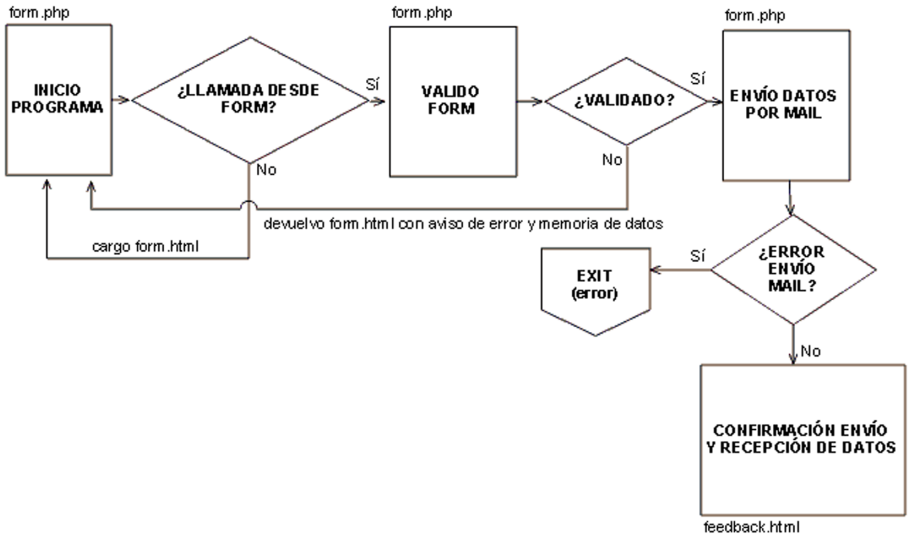
El código de la aplicación se distribuye en tres ficheros:

- **form.php**. El *script* de PHP principal que carga el formulario HTML, valida sus datos y los envía por correo a su destinatario mediante una cuenta Gmail.

- **form.html**. El archivo con el formulario HTML.

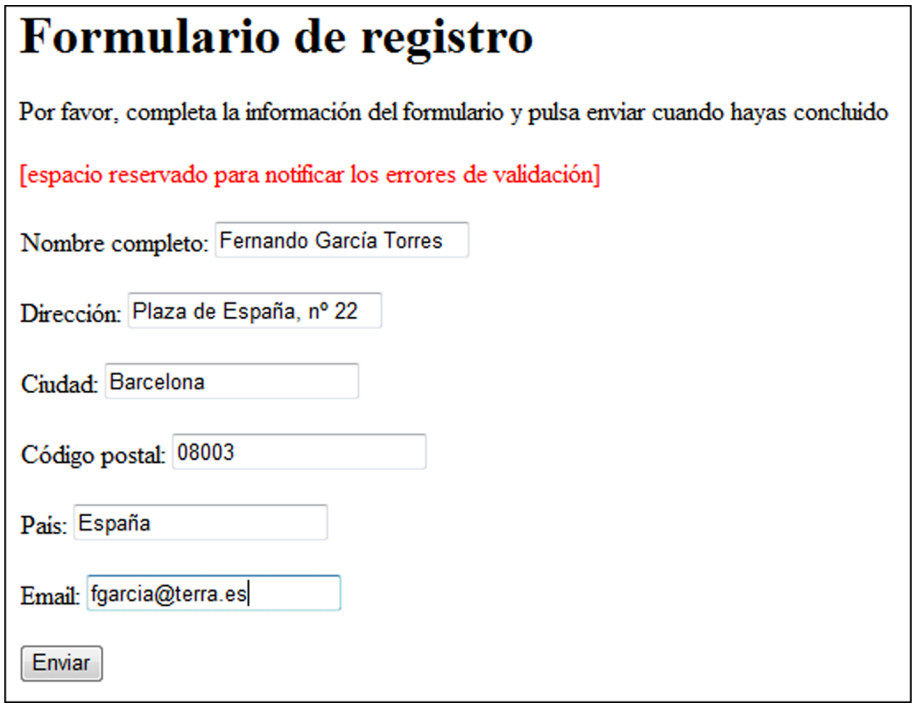
- **feedback.php**. Una sencilla página web con la confirmación de envío y recepción de los datos.

Lógica de aplicación:



Inicio del programa

El programa se arranca desde el *script* **form.php**. Si no existe una llamada desde el formulario (si no se ha pulsado el botón "Enviar"), se carga el formulario vacío (**form.htm**) por primera vez.



En el ejemplo particular, la validación del formulario tendrá en cuenta las siguientes reglas:

- Todos los campos son de dato obligatorio (no pueden estar vacíos).
- El valor de código postal debe ser numérico (desde 00001 a 52999).
- El valor del correo electrónico debe ser sintácticamente correcto.

Si no se cumple la validación, se devuelve el formulario con la notificación del error y la memoria de datos.

Si la validación de los datos del formulario es correcta, éstos se envían por correo mediante la clase phpmailer.

Finalmente, el usuario recibe la confirmación del envío.

Fichero principal (form.php)

```
<?php

// LLAMADA DESDE FORMULARIO
if(isset($_POST['submit']) ) {

# Preparo datos pasados desde el form
$FormData = array();

# Limpio espacios en blanco y caracteres de escape
foreach($_POST as $key=>$value) {
   $FormData[$key] = stripslashes(trim($value));
}
```

```
// VALIDO FORM
# Códigos de error
# 0 - sin error
# 1 - datos requeridos
# 2 - código postal incorrecto
# 3 - mail sintaxis incorrecta

$error_frm = 0;

# TEST 1: datos requeridos
if(empty($FormData['nombre']) || empty($FormData['direccion']) ||
empty($FormData['ciudad']) || empty($FormData['cod_postal'])
   || empty($FormData['pais']) ||  empty($FormData['email']) ) {
   $error_frm = 1;
}


# TEST 2: valido código postal con una expresión regular
# Los códigos postales en España van desde 00001 al 52999
if(!$error_frm && !preg_match("/^([1-9]{2}|[0-9][1-9]|[1-9][0-9])
[0-9]{3}$/", $FormData['cod_postal']) ) {
   $error_frm = 2;
}

# TEST3: valido email con una expresión regular
if(!$error_frm && !preg_match("/^[A-z0-9\._-]+@[A-z0-9][A-z0-9-]*
(\.[A-z0-9_-]+)*\.([A-z]{2,6})$/", $FormData['email']) ) {
   $error_frm = 3;
}
```

```
// PASADOS TODOS LOS TESTS ---
if(!$error_frm){

   // ENVÍO DATOS MEDIANTE UNA CUENTA GMAIL

   #  Cargo clase phpmailer
   include_once('phpMailer_v2.3/class.phpmailer.php');
   # Creo una nueva instancia
   $mail = new PHPMailer();

   # Servicio Gmail
   # Le digo que voy a usar SMTP (el modo de envío)
   $mail->IsSMTP();
   # GMail usa SSL/TLS como protocolo de comunicación/autenticación
   $mail->Host = 'ssl://smtp.gmail.com';
   # El Puerto de comunicación del servidor SMTP de Gmail
   $mail->Port = 465;
   # Activo la autentificación SMTP
   $mail->SMTPAuth = true;
   # Mi nombre de usuario en Gmail.
   $mail->Username = 'usuario@gmail.com';
   # Mi contraseña en Gmail
   $mail->Password = '*******';

   # Propiedades del Mensaje
   # Codificación de caracteres
   $mail->CharSet = "UTF-8";
   # mail y nombre del remitente
   $mail->From = $FormData['email'];
   $mail->FromName = $FormData['nombre'];
   # Mail del destinatario
   $mail->AddAddress('usuario@gmail.com');
   # Puedo añadir otro destinatario más
   $mail->AddAddress('usuario@uoc.edu');

   # El tema del mensaje
   $mail->Subject = "Formulario de registro";
   # Mail en formato HTML
   $mail->IsHTML(true);
   # Caracteres por línea
   $mail->WordWrap = 50;
   # Cuerpo del mensaje

$cuerpo = "
FORMULARIO DE REGISTRO \n
Fecha de registro:".date(d."-".m."-".Y)."\n
-----------------------------------------------\n
Nombre: ".$FormData['nombre']."\n
Dirección: ".$FormData['direccion']."\n
Ciudad: ".$FormData['ciudad']."\n
Código Postal: ".$FormData['cod_postal']."\n
País: ".$FormData['pais']."\n
Email: ".$FormData['email']."\n
-----------------------------------------------\n
";
   # Cargo el contenido del mensaje con los saltos de linea convertidos en <br />
   $mail->Body=nl2br(htmlentities($cuerpo, ENT_COMPAT));
   # También cargo el contenido del mensaje con formato de texto plano
   $mail->AltBody = $cuerpo;
```

```
   // ENVÍO MENSAJE
   # Send Mail
   if(!$mail->Send()) {
   echo "El mensaje no puede ser enviado. <p>";
      echo "phpMailer Error: " . $mail->ErrorInfo;
         exit;
   }


   // CONFIRMACIÓN DE ENVÍO Y RECEPCIÓN DE DATOS ---
   include_once('feedback.html');
   exit;

   }
   else {
      # Devuelvo form con memoria de datos y el aviso de error de validación
      include_once('form.html');
   }

} else
   include_once('form.html');

?>
```

Formulario HTML (form.html)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
   <title>Formulario para enviar los datos a una cuenta Gmail</title>
</head>


<body>
  <Hl>Formulario de registro</Hl>
      <p>Por favor, completa la información del formulario y pulsa enviar
      cuando hayas concluido</p>
  <!-- espacio reservado para notificar los errores de validación -->
      <?php if($error_frm == 1) { ?>
            <p style="color: red;">
              ERROR: todos los campos son de dato requerido
            </p>
      <?php } elseif($error_frm == 2) { ?>
       <p style="color: red;">ERROR: el código postal no es correcto</p>
      <?php } elseif($error_frm == 3) { ?>
       <p style="color: red;">ERROR: el email no es correcto</p>
      <?php } ?>


<form action="form.php" method="post">
   <p>Nombre completo:
      <label>
<input type="text" name="nombre" id="nombre" value="<?=$FormData['nombre']?>" />
      </label>
   </p>
```

```
   <p>Dirección:
      <label>
         <input type="text" name="direccion" id="direccion"
         value="<?=$FormData['direccion']?>" />
      </label>
   </p>

   <p>Ciudad:
      <label>
         <input type="text" name="ciudad" id="ciudad" value="<?=$FormData['ciudad']?>" />
      </label>
   </p>

   <p>Código postal:
      <label>
         <input type="text" name="cod_postal" id="cod_postal"
         value="<?=$FormData['cod_postal']?>" />
      </label>
   </p>

   <p>País:
      <label>
         <input type="text" name="pais" id="pais" value="<?=$FormData['pais']?>" />
      </label>
   </p>

   <p>Email:
   <label>
         <input type="text" name="email" id="email" value="<?=$FormData['email']?>" />
      </label>
   </p>
   <p>
      <input type="submit" name="submit" id="submit" value="Enviar" />
   </p>
</form>

</body>
</html>
```

Confirmación de envío (feedback.html)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
       <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
       <title>Documento sin t&iacute;tulo</title>
</head>
```

```
<body>
	<h1>Formulario de registro</h1>
	<hr />
	<p>Apreciado <?=$FormData['nombre']?></p>
	<p>Los datos de su solicitud se han recibido correctamente.</p>
	<p>
		La matrícula se formalizará una vez que se compruebe el pago de su
		inscripción.
	</p>
	<p>
		Le tendremos informado mediante su cuenta de correo <a
		href="mailto:<?=$FormData['email']?>"><?=$FormData['email']?></a>
	</p>
</body>
</html>
```

4. El formulario anterior con código de seguridad anti *spambots* (captcha)

Captcha es el acrónimo de *Completely Automated Public Turing test to tell Computers and Humans Apart* (prueba de Turing pública y automática para diferenciar máquinas y humanos).

Los captchas son utilizados para evitar que robots, también llamados *spambots*, puedan utilizar ciertos servicios. Por ejemplo, para que no puedan participar en encuestas, registrarse para usar cuentas de correo electrónico (o su uso para envío de correo basura) o, más recientemente, para evitar que correo basura pueda ser enviado por un robot (el remitente debe pasar el test antes de que se entregue al destinatario).

La idea básica para crear un captcha es generar primero un texto aleatorio; luego, basado en este texto, generamos una imagen que se muestra al usuario, y, finalmente, se compara el texto ingresado por el usuario con la palabra aleatoria generada.

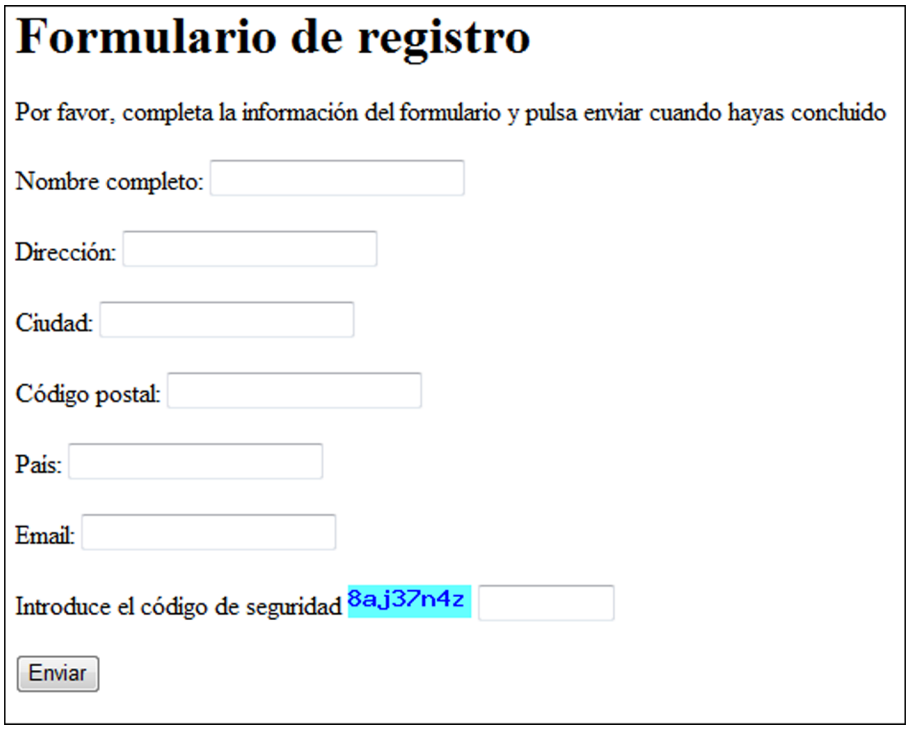
Generar el captcha (captcha.php)

El primer paso consiste en crear una imagen con un texto aleatorio mediante el *script* **captcha.php**, el cual creará automáticamente el texto y salvará su valor en la variable de sesión $_SESSION['captcha'].

Lo siguiente es mostrar la imagen generada en el formulario HTML para que el usuario pueda leerlo e ingresar el texto para ser verificado. Para ello, editamos el archivo **form.html** y añadimos al final las siguientes líneas de código.

```
<p>
      Introduzco el código de seguridad
      <img src="captcha.php" alt="" width="75" height="20" vspace="1" align="top" />
      <input name="captcha" type="text" id="captcha" size="8" maxlength="10" />
</p>
```

El formulario HTML con el captcha tendrá el siguiente aspecto:



El paso siguiente consiste en verificar el texto ingresado en el formulario y compararlo con la variable de sesión que contiene el texto generado aleatoriamente. Para ello, será necesario hacer algunas modificaciones en el programa principal **form.php**.

Al comienzo del programa inicializamos la sesión para poder leer el dato:

```
<?php
        session_start();
        // LLAMADA DESDE FORMULARIO
        if(isset($_POST['submit']) ) {
        ...
        ...
?>
```

En la sección de validación del formulario añadimos un nuevo test con el que comparamos el texto ingresado con el texto que tenemos en la variable de sesión.

```
...
...
// VALIDO FORM
# Códigos de error
# 0 - sin error
# 1 - datos requeridos
# 2 - código postal incorrecto
# 3 - mail sintaxis incorrecta
# 4 - código de seguridad incorrecto
...
...
# TEST4: valido captcha
 if(!$error_frm && $_SESSION['captcha'] != $FormData['captcha']) {
    $error_frm = 4;
 }
...
...
```

Finalmente, en la parte inicial de **form.html** añadimos el mensaje de error de validación del captcha.

```
<?php } elseif($error_frm == 4) { ?>
       <p style="color: red;">ERROR: código de seguridad incorrecto</p>
<?php } ?>
```

Fichero generador del captcha (captcha.php)

```
<?php
// Crear CAPTCHA con PHP

# Creo una sesión con PHP
session_start();
# Salvo en una variable de sesión el valor del captcha
$_SESSION['captcha'] = randomText(8);


# Genero la imagen con las funciones de la librería GD de PHP
$img = imagecreate(75, 20);
$fondo = imagecolorallocate($img, 100, 255, 255);
$color_texto = imagecolorallocate($img, 0, 0, 255);
imagestring($img, 5, 0, 0, $_SESSION['captcha'], $color_texto);


# Envío la imagen tras las cabeceras HTTP
header("Content-type: image/png");
imagepng($img);
exit;
```

```
# Genero el valor del captcha
function randomText($length) {
    $pattern = "1234567890abcdefghijklmnopqrstuvwxyz";
    for($i=0;$i<$length;$i++) {
      $key .= $pattern{rand(0,35)};
    }
    return $key;
}
?>
```

5. Geolocalización con GeoIp y Google Maps

El geoposicionamiento por IP tiene interesantes beneficios en aplicaciones de blog, Chat, foros, seguridad, *urban mapping*, etc. Por ejemplo, para combatir prácticas indeseables como el *phishing* en el desarrollo de aplicaciones distribuidas en entornos abiertos y heterogéneos.

¿Cómo se realiza el geoposicionamiento de visitantes?

1) Se obtiene la IP del visitante.

2) Se averigua la *IP Address Look Up* de esa IP para saber su situación. Para esto utilizamos la API de un servicio de *Geolocation by IP Address* para acceder a la información geográfica de sus bases de datos.

3) Se genera el mapa con la API de Google Maps, geoposicionando dentro del mapa la información obtenida con el paso anterior.

Existen en Internet multitud de servicios de *Geolocation by IP Address*, comerciales y gratuitos, con los que es posible averiguar la *IP Address Look Up* de los usuarios con información geográfica del tipo: país, región, ciudad, latitud, longitud, ZIP *code*, *time zone*, velocidad de conexión, ISP, etc.

Productos comerciales realizados con el respaldo de bases de datos de pago son, por ejemplo:

- IP2Location™
- Geobytes
- MaxMind, GeoIP

Productos hechos con el respaldo de bases de datos de acceso gratuito son:

- HostIP.info
- NetGeo - The Internet Geographic Database

Tanto los productos comerciales como gratuitos ponen a disposición de los usuarios una API para localizar direcciones IP con información geográfica compatible para diferentes lenguajes. Por ejemplo, MaxMind ofrece su *Open Source binary API* para los lenguajes C, Perl, Apache, Java, Python, Ruby, etc. En el caso de hostip.info hay una API para localizar direcciones IP con información geográfica que nos permite hacer, entre otras cosas, saludar a los visitantes con la bandera del país desde donde se conectan.

Casi todos estos productos tienen en su página web un localizador *on-line*. Los hay que incluso posicionan el resultado de nuestra búsqueda en Google Maps; por ejemplo:

- Geo IP Tool
- Ip-adress
- Seomoz

Por norma general, la información de los productos con respaldo de bases de datos de acceso gratuito no es excesivamente completa. Muchas IP no quedan localizadas y de algunas ciudades no disponen de la latitud y la longitud (con lo que hay que excluirlas y no se pueden situar en el mapa).

Los productos con respaldo de bases de datos de pago requieren una licencia de uso que debe renovarse cada cierto tiempo. Las bases de datos licenciadas se actualizan en línea en nuestro servidor tras su instalación y una vez registrados. Por ejemplo, MaxMind ofrece para su producto GeoIP, ocho bases de datos de diferentes propósitos. Afortunadamente para los desarrolladores web como nosotros, MaxMind tiene dos versiones gratuitas de sus bases de datos GeoIP Country y GeoIP City, aunque, como es lógico, hay limitaciones en el volumen de información y sobre el grado de exactitud de los datos de país y ciudad.

Estas dos bases de datos se pueden descargar de Maxmind, junto con el código fuente o binario de GeoIP, y la API de PHP, para así instalar la aplicación en un servidor de Internet.

5.1. Acceso a la base de datos GeoIP para resolver la *IP Address Look Up* de una IP

Una vez instalada la base de datos GeoIP y la extensión de GeoIP de PHP, ya es posible obtener la *IP Address Look Up* con la ayuda del siguiente *script* (**geoip.php**), que recibirá la IP que debe resolver como parámetro de la URL: http://host/geoip.php?ip=213.73.38.93.

Este *script* devolverá un texto simple con la información de la consulta encadenada con un formato especial, tal y como se ve en el siguiente ejemplo:

```
VERSION:GEO-106FREE 20080101 Build 1 Copyright (c) 2007 Max-
Mind LLC
&COUNTRY:Spain&CITY:Barcelona&LAT:41.3833007812&LONG:2.18330001831
```

Programa: geoip.php

```
<?php

// Compruebo que se ha pasado el parámetro por la URL
if(!isset($_GET['ip']) || empty($_GET['ip'])) {
    echo "ERROR: no se pasó la IP en el URL";
    exit;
}

// Compruebo con una expresión regular que la IP es correcta
if(!preg_match( "/^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$/", $_GET['ip']) ) {
    echo "ERROR: la dirección IP no es correcta";

}

// GeoIP IP Address Look Up
$GeoIpData = array();

# Información sobre la versión instalada de la base de datos GeoIP
$GeoIpData['VERSION'] = geoip_database_info(GEOIP_COUNTRY_EDITION)."\n";
# País
$GeoIpData['COUNTRY'] = geoip_country_name_by_name($_GET['ip']);
# Ciudad, latitud, longitud
$datos = geoip_record_by_name($_GET['ip']);
$GeoIpData['CITY'] = $datos['city'];
$GeoIpData['LAT'] = $datos['latitude'];
$GeoIpData['LONG'] = $datos['longitude'];

# Junto los elementos del array en una cadena de texto con formato (nombre:valor)
$elementos = array();
while (list($key, $value) = each($GeoIpData))  {
    $elementos[] = $key.":".$value;
}

# Encadeno los elementos con el signo &
$resultado = implode("&", $elementos);

# Devuelvo el resultado como flujo de datos tras una cabecera HTTP
header('Content-Type: text/plain');
echo $resultado;
exit;

?>
```

5.2. Geoposicionar en un mapa de Google la procedencia geográfica del visitante

La aplicación **geolocacion.php** muestra en un mapa de Google Maps la ubicación geográfica de la persona que visita la página web. Este programa le pasa a **geoip.php** una IP de la que queremos obtener información geográfica, y este último le retorna los datos de la consulta de la base de datos GeoIP. La IP que se debe geolocalizar se puede obtener automáticamente por medio de la variable $_SERVER['REMOTE_ADDR'], o introducir manualmente como parámetro de la URL: http://host/geolocacion.php?ip=213.73.38.93

Para poder utilizar el servicio de Google Maps será necesario darse de alta:

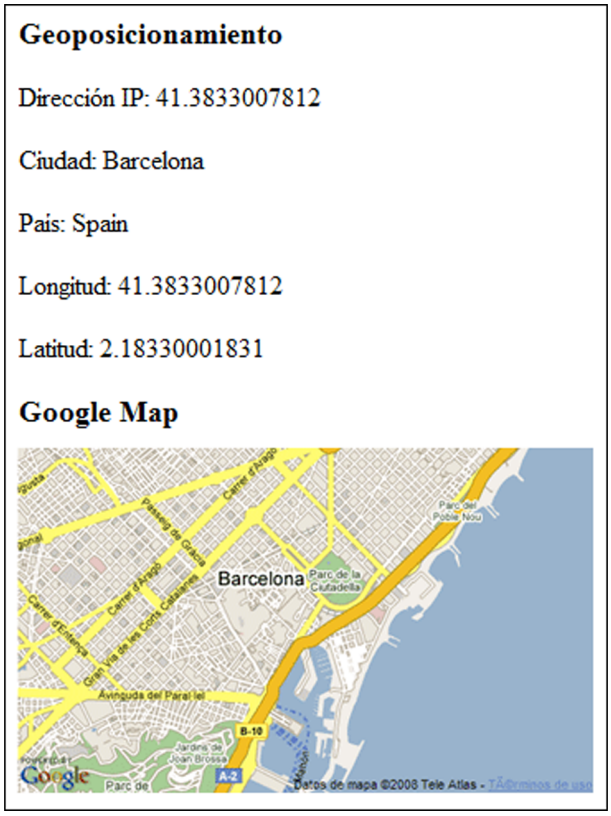
1) El registro se efectúa desde la página Google Maps y es gratuito.

2) Habrá que especificar la URL y el directorio de la página que usará el servicio de mapas de Google.

3) El proceso termina con la obtención de una clave de API de Google Maps que deberemos añadir, más tarde, al objeto <script> que nos mostrará el mapa.

4) Las coordenadas geográficas LAT y LONG obtenidas con el *script* de PHP deberán ir colocadas en la siguiente línea de código de la API de Google Maps:

```
map.setCenter(new GLatLng(37.4419, -122.1419), 13);
```



Programa principal (geolocacion.php)

```
<?php

// IP pasada manualmente de la URL
# p.e. http://host/geolocation.php?ip=213.73.38.93
$ip = "";
if(isset($_GET['ip']) && !empty($_GET['ip']) ) {

   # Compruebo con una expresión regular que la IP es correcta
      if(preg_match("/^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$/", $_GET['ip']) )
      {
         $ip = $_GET['ip'];
   }
}
```

```
// IP automática obtenida automáticamente
if(empty($ip) )  {
   $ip =  $_SERVER['REMOTE_ADDR'];
}

// Consulto la base de datos para resolver la IP Address Look Up
$geoIpURL = "http://host/geoip.php?ip=$ip";

# Abro el contenido del URL para sólo lectura
if($GeoIpFP = fopen($geoIpURL,r)) {

   # Habilito el uso de búferes de salida
   ob_start();

   # Leo el flujo de datos y lo cargo en el búfer de salida
   fpassthru($GeoIpFP);

   # Cargo en una variable los contenidos del búfer de salida, sin borrarlo
   $GeoIpQueryData = ob_get_contents();

   # Deshabilito los búferes de salida
   ob_end_clean();

   # Cierro el apuntador al URL abierto
   fclose($GeoIpFP);

   # Cargo las parejas de datos en un array
   $parejas = array();
   $parejas = explode('&', $GeoIpQueryData);

   # Cargo los datos de cada pareja en un array asociativo
   $GeoIPData = array();
   for($i=0; $i < count($parejas); $i++) {
      list($k, $v) = split(":",$parejas[$i]);
      $GeoIPData["$k"] = $v;
   }

}
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
   <title>Geoposicionamiento</title>

      <script src="http://maps.google.com/maps?file=api&amp;v=2&amp;key=ABQIAAAAvqU-
      yHDNLzzic0kICu0UmBRA4kvN07udzPbkMjf7Gs6xqBycohQgma4w6SDtmk4tjEo4x0HuIpV96w"
      type="text/javascript"></script>

      <script type="text/javascript">
         //<![CDATA[
         function load() {
            if (GBrowserIsCompatible()) {
               var map = new GMap2(document.getElementById("map"));
               map.setCenter(new GLatLng(<?=$GeoIPData['LAT']?>, <?=$GeoIPData['LONG']?>),
               13);
            }
         }
         //]]>
      </script>
</head>
```

```
<body onload="load()" onunload="GUnload()">
   <h3>Geoposicionamiento</h3>
   <p>Dirección IP: <?=$GeoIPData['LAT']?></p>
   <p>Ciudad: <?=$GeoIPData['CITY']?></p>
   <p>País: <?=$GeoIPData['COUNTRY']?></p>
   <p>Longitud: <?=$GeoIPData['LAT']?></p>
   <p>Latitud: <?=$GeoIPData['LONG']?></p>
   <h3>Google Map</h3>
   <div id="map" style="width: 500px; height: 300px"></div>
</body>
</html>
```

Recurso: Módulo 4. Desarrollo web con PHP y MySQL. Descripción: Este es el cuarto módulo del recurso "Laboratorio de PHP y MySQL", en el que se analizan desarrollos web con php y MySQL. Idioma: ES Categoría: Servicios de Internet Fecha de alta: 2010-06-17 00:00:00.0