

# Tema: Creación de Formularios HTML

## 1.- Introducción

La Web se ha convertido en una poderosa herramienta para las empresas que se dedican a realizar encuestas. En la actualidad es muy común acceder a páginas que nos ofrecen la información o el servicio demandado, siempre que el “navegante” proporcione cierta información: p.e. datos personales (¡ojo con la privacidad!), gustos, preferencias, o simplemente la opinión sobre alguna cuestión puntual (los diarios online suelen realizar encuestas sobre temas de actualidad). Con este propósito los formularios han sido una de las herramientas que han ayudado a este auge.

En este capítulo trataremos de describir el modo de realizar formularios HTML, en el supuesto de que el lector no está familiarizado con este lenguaje. Para ello, definiremos las cuestiones básicas del lenguaje, a medida que se necesiten ciertos conceptos. Además, acompañaremos estos conceptos con ejemplos en lenguaje HTML, con idea de que el lector copie el código en un fichero, que grabado con extensión html (o htm) le permita visualizar en qué consiste la orden descrita.

Como cuestión previa diremos que el lenguaje se compone de un conjunto de sentencias denominadas “tags” que permiten delimitar zonas de la página web en las que el “programador” define una acción. Por ejemplo, si quisieramos poner en negrita una zona del texto, escribiríamos:

```
<b>texto en negrita</b>
```

Los tags a que nos referimos son `<b>` y `</b>` y es inmediato observar que el primero define el comienzo de la acción y el segundo dónde finaliza la misma. En el caso de un formulario, los tags que lo especifican son:

```
<form>.....</form>
```

y dentro de ellas se recogerán todas las variables de entrada.

Los formularios nos van a permitir, desde dentro de una presentación web, solicitar información al visitante. La información que se solicita es de tipos diversos (p.e. escribir el nombre, marcar una casilla si está casado, seleccionar la marca de su vehículo dentro de una lista, etc.), y por tanto habrá distintas opciones, atributos y modificadores que permitan definir todos esos tipos. De este modo, a la tag de apertura `<form>` le acompañarán estos atributos:

`action=""` Entre comillas se indica el programa que va a tratar las variables enviadas con el formulario, un guión CGI o la URL mailto.

`Method=""` Indica el método de transferencia de las variables. Post, si se envía a través del STDIO. Get, si se envía a través de la URL.

A continuación mostramos un ejemplo:

```

<HTML>
<HEAD>
<TITLE>Ejemplo 1 de formulario</TITLE>
</HEAD>
<BODY>

<H1>Formularios</H1>

<FORM ACTION="mailto:unaprueba" METHOD="POST">
<INPUT TYPE="text" NAME="nombre"><BR>
<INPUT TYPE="submit"><INPUT TYPE="Reset">
</FORM>

</BODY>
</HTML>

```

El resultado es:

The screenshot shows a window titled "Formularios". Inside the window, there is a single input field at the top. Below it are two buttons: "Enviar consulta" on the left and "Restablecer" on the right. The window has scroll bars on the right side.

## 2.- Comandos básicos

Los diversos campos de entrada de datos en los formularios estándar son:

### <INPUT>

La tag <input> define la introducción de variables. Junto a esta tag encontraremos los siguientes atributos:

**type=""** Indicará el tipo de variable a introducir.

**text** Indica que el campo a introducir será un texto. Sus atributos:

**maxlength=""** Seguido de un valor que limitará el número máximo de caracteres a introducir en ese campo.

**size=""** Seguido de un valor que limitará el numero de caracteres a mostrar en pantalla.

**value=""** Indica que no hay valor inicial del campo.

**Password** Indica que el campo a introducir será una palabra de paso. Mostrará asteriscos en lugar de letras escritas. Sus atributos serán los mismos que para **text**.

**Checkbox** El campo se elegirá marcando de entre varias opciones una casilla cuadrada.

**value=""** Entre comillas se indicará el valor de la casilla.

**checked** La casilla aparecerá marcada por defecto.

**Radio** El campo se elegirá marcando de entre varias opciones una casilla circular.

value="" Entre comillas se indicará el valor de la casilla.

Image El campo contendrá el valor en coordenadas del punto de la imagen que haya pinchado. Atributo obligatorio:

src="" Entre comillas escribiremos el nombre del archivo de imagen.

hidden El visitante no puede modificar su valor ya que no está visible. Se manda siempre junto al atributo value= seguido de su valor entre comillas.

Name="" Indicará el nombre que se asigna a un determinado campo.

```
<HTML>
<HEAD>
<TITLE>Ejemplo 2 de formularios</TITLE>
</HEAD>
<BODY>

<H1>Formularios</H1>

<FORM ACTION="mailto:unaprueba" METHOD="POST">
Texto: <INPUT TYPE="text" NAME="nombre"><BR>
Password: <INPUT TYPE="password" NAME="contra"><BR>
Sexo:<INPUT TYPE="radio" NAME="boton1" VALUE="1"> Hombre
<INPUT TYPE="radio" NAME="boton1" VALUE="2">Mujer<BR>
Vehiculo:<INPUT TYPE="checkbox" NAME="Moto" VALUE="Si">Moto
<INPUT TYPE="checkbox" NAME="Coche" VALUE="" CHECKED>Coche

<BR><BR>
<INPUT TYPE="submit"><INPUT TYPE="Reset">
</FORM>

</BODY>
</HTML>
```

El resultado es:

The screenshot shows a web page with the title "Formularios". The form includes fields for "Texto" (text input), "Password" (password input), "Sexo" (radio buttons for "Hombre" and "Mujer", with "Hombre" selected), "Vehiculo" (checkboxes for "Moto" and "Coche", with "Coche" checked), and two buttons at the bottom: "Enviar consulta" and "Restablecer".

## <SELECT>

Las tags <select>.....</select> encierran los valores que podremos elegir a partir de una lista. Los atributos que acompañan a la tag de apertura son:

name="" Indicará el nombre del campo de selección.

Size="" Indicará el número de opciones visibles. Si le asignamos 1, la selección se presentará como un menú desplegable. Si le asignamos un valor mayor se presentará como una lista con barra de desplazamiento.

multiple Indica si se pueden realizar multiples selecciones.

Las diferentes opciones de la lista se indicarán mediante la tag <option> que puede acompañarse del atributo selected para indicar cual es la opción que aparecerá por defecto. Si no lo especificamos, siempre será la primera de la lista.

```
<HTML>
<HEAD>
<TITLE>Ejemplo 3</TITLE>
</HEAD>
<BODY>

<H1>Formularios</H1>

<FORM ACTION="mailto:unaprueba" METHOD="POST">
<SELECT NAME="Colores" MULTIPLE>
<OPTION VALUE="r">Rojo</OPTION>
<OPTION VALUE="g">Verde</OPTION>
<OPTION VALUE="b">Azul</OPTION>
</SELECT>
<BR><BR>
<SELECT NAME="Colores" SIZE="1">
<OPTION VALUE="r">Rojo</OPTION>
<OPTION VALUE="g">Verde</OPTION>
<OPTION VALUE="b">Azul</OPTION>
</SELECT>
<BR><BR>
<INPUT TYPE="submit"><INPUT TYPE="Reset">
</FORM>

</BODY>
</HTML>
```

Se obtiene la página siguiente:

# Formularios

The screenshot shows a web page with a title 'Formularios'. Below the title is a dropdown menu with three options: 'Rojo', 'Verde', and 'Azul'. The 'Azul' option is currently selected. Below the dropdown is a button labeled 'Rojo' with a dropdown arrow. At the bottom of the form are two buttons: 'Enviar consulta' and 'Restablecer'.

## <TEXTAREA>

Con las tags <textarea>.....</textarea> definimos un texto de múltiples líneas para que el visitante pueda incluir un comentario junto a sus datos.

Junto a la tag de apertura pueden aparecer los siguientes atributos:

name="" Nombre del campo.  
Cols="" Numero de columnas de texto visible.  
Rows="" Numero de filas de texto visible.

```
<HTML>
<HEAD>
<TITLE>Ejemplo 4</TITLE>
</HEAD>
<BODY>

<H1>Formularios</H1>

<FORM ACTION="mailto:unaprueba" METHOD="POST">
<TEXTAREA COLS=20 ROWS=10 NAME="Texto">
</TEXTAREA>
<BR><BR>
<INPUT TYPE="submit"><INPUT TYPE="Reset">
</FORM>

</BODY>
</HTML>
```

Con este código se obtiene:

# Formularios



## <BUTTON>

A partir de la implementación de los estándares HTML 4.0 contamos con varias etiquetas nuevas para construir formularios, siendo BUTTON una de ellas, de bastante utilidad, aunque no compatible con algunas versiones de Nestcape. Las tags <BUTTON>...</BUTTON> proporcionan un método único para la implementación de cualquier tipo de botón de formulario. Sus principales atributos son:

type= " tipo ", que puede tomar los ya conocidos valores submit (por defecto), reset y button.

name= " nombre ", que asigna un nombre identificador único al botón.

value= " texto ", que define el texto que va a aparecer en el botón.

La principal ventaja que aporta estas etiquetas es que ahora vamos a poder introducir dentro de ellas cualquier elemento de HTML, como imágenes y tablas.

```
<HTML>
<HEAD>
<TITLE>Ejemplo 5</TITLE>
</HEAD>
<BODY>

<H1>Formularios</H1>
<FORM ACTION="mailto:unaprueba" METHOD="POST" ENCTYPE="text/plain"
      name="miform">

<BUTTON name="boton_1" type="button">
<TABLE width="10" cellspacing="0" cellpadding="2" border="1">
<TR>
<TD>uno</TD>
```

```

<TD>dos</TD>
</TR>
<TR>
<TD>tres</TD>
<TD>cuatro</TD>
</TR>
</TABLE>
</BUTTON>
</FORM>
</BODY>
</HTML>

```

Con lo que se obtiene:

## Formularios

uno	dos
tres	cuatro

En el ejemplo siguiente se habrá de incluir una imagen de nombre “pajaro.gif” a la carpeta imágenes:

```

<HTML>
<HEAD>
<TITLE>Ejemplo 6</TITLE>
</HEAD>
<BODY>

<H1>Formularios</H1>
<FORM ACTION="mailto:unaprueba" METHOD="POST" ENCTYPE="text/plain"
name="miform">

<BUTTON name="boton_1" type="button">
<IMG src="imagenes/pajaro.gif" width="75" height="30" border="0" alt="enviar">
</BUTTON>
</FORM>
</BODY>
</HTML>

```

### <LABEL>

Hasta hora, el texto que acompañaba a los campos de entrada no estaba asociado a los mismos de ninguna manera. Así, por ejemplo, si pulsamos en el texto que acompañaba a un control de confirmación, no sucedía nada. Ahora, en cambio, si utilizamos la etiqueta

LABEL, el control cambiara de estado (disponible en versiones superiores de Netscape).

```
<HTML>
<HEAD>
<TITLE>Ejemplo 8</TITLE>
</HEAD>
<BODY>

<H1>Formularios</H1>
<FORM ACTION="mailto:unaprueba" METHOD="POST" ENCTYPE="text/plain"
name="miform">

<LABEL>
<INPUT type="checkbox" name="correo">
deseo que me envíen correo
</LABEL>
</FORM>
</BODY>
</HTML>
```

## <FIELDSET>

Hasta ahora, no disponíamos de ninguna manera de agrupar visualmente varios controles, a menos que usemos elementos que no son del formulario, como tablas o imágenes. Si encerramos una parte de un formulario dentro de la etiqueta FIELDSET se mostrara un rectángulo alrededor de los mismos.

Podemos indicar un título por medio de la etiqueta LEGEND, que admite el parámetro align="left / center / right / top /bottom", lo que nos permite alinear el título horizontal y verticalmente. Además, deberemos introducir el conjunto en una celda de tabla con un ancho determinado, ya que de no hacerlo así el recuadro abarcará todo el ancho de pantalla disponible.

```
<HTML>
<HEAD>
<TITLE>Ejemplo 9</TITLE>
</HEAD>
<BODY>

<H1>Formularios</H1>
<FORM ACTION="mailto:unaprueba" METHOD="POST" ENCTYPE="text/plain"
name="miform">
<TABLE width="200">
<TR>
<TD>
<FIELDSET>
<LEGEND align="left"><font color="red">Caja de texto</font></LEGEND>
pon tu nombre:
```

```

<INPUT type="text" size="15">
</FIELDSET>
</TD>
</TR>
</TABLE>
</FORM>
</BODY>
</HTML>

```

## Formularios

Caja de texto  
pon tu nombre:

### 3.- Mejoras de formularios con Javascript

Una de las herramientas más potentes para introducir mejoras en la definición de formularios es el lenguaje Javascript. Es un lenguaje formado por trozos de código que se insertan en las propias páginas web, y que permite el manejo dinámico de las mismas por parte del programador, de tal forma que es posible verificar y/o cambiar propiedades de los elementos de la página sin tener que volver a realizar una conexión con el servidor, así como realizar cálculos y transferencias de datos entre los campos del formulario.

Puesto que el objetivo de este capítulo no es describir el lenguaje Javascript, sino mostrar que mejoras puede proporcionar a un formulario, nos limitaremos a presentar algunos ejemplos con su código correspondiente. Se puede acceder al código en el directorio de Descargas.

#### Modificar cajas de texto

- Ejemplo 1 (fichero valid\_campo1.txt): Comprobar que el usuario no deje el campo en blanco. Se trata de lanzar un mensaje de error si se pretende enviar el formulario con la caja de texto sin llenar. Además devuelve el foco a la misma.
- Ejemplo 2 (fichero valid\_campo2.txt): Pasa a mayúsculas el texto introducido.
- Ejemplo 3 (fichero valid\_campo3.txt): Quitar los espacios en blanco que se hayan introducido en la caja antes del envío (ya que estos pueden dar problemas al programa CGI).
- Ejemplo 4 (fichero valid\_campo4.txt): Script que sólo permite la entrada de números. Esto será especialmente útil para campos como códigos postales, cuestiones numéricas, etc.
- Ejemplo 5 (fichero valid\_campo5.txt): Validar que sólo se introduzcan números sin decimales o con dos decimales.

## **Modificar checkbox y buttons**

- Ejemplo 6 (fichero valid\_opcion1.txt): Se obliga al usuario a que marque sólo un número determinado de opciones dentro de un conjunto de ellas.
- Ejemplo 7 (fichero valid\_opcion2.txt): Si nos interesa, los checkbox también pueden ser autoexcluyentes, pero sólo es conveniente su uso en casos como el de este ejemplo, en el que obligamos al usuario a elegir entre una opción que selecciona todas las demás o marcar las que desea una a una.
- Ejemplo 8 (fichero valid\_opcion3.txt): Asocia un mensaje de alerta al radio botón, de tal forma que cuando el usuario pinche en él se active este mensaje de advertencia.

## **Validación de formularios**

Se trata de verificar, someramente, los datos que se introducen en el cuestionario

- Ejemplo 9 (fichero validación1.txt): Este formulario requiere que todos los campos sean completados. Si queda alguno vacío se lanza una ventana de aviso, requiriendo al visitante para que lo complete. Si todos los campos son escritos se lanza una ventana de comprobación, tras la que se envían los datos al servidor.
- Ejemplo 10 (fichero validación2.txt): Formulario de elecciones de grupos a base de radio botones, de forma que se asegura no deja ninguna opción sin elegir.
- Ejemplo 11 (fichero validación3.txt): El siguiente ejemplo muestra un formulario en el que los campos van apareciendo sucesivamente uno tras otro, almacenándose la información de cada uno de ellos hasta el envío final. En botón de envío está desactivado.
- Ejemplo 11 (fichero validación4.txt): Es muy útil a veces poder desactivar y/o activar elementos de formulario cuando nos convenga. Un ejemplo típico sería el caso de establecer ciertos campos que sólo se puedan llenar si antes se cumple alguna condición. Como enumerar una a una todas las posibles combinaciones sería muy largo, en el siguiente ejemplo podemos ver (y obtener de su código) cómo se activan y desactivan numerosos tipos de campo, siendo posible hacerlo elemento a elemento o todos a la vez.
- Ejemplo 12 (fichero validación5.txt): En este ejemplo se van activando distintas partes del cuestionario.
- Ejemplo 13 (fichero valid\_test1.txt): El siguiente script nos muestra cómo crear un sencillo test, muy fácil configurar.