

Uso de selectores CSS avanzados

Selectores CSS avanzados

- › Utilizando solamente los selectores básicos vistos en la clase anterior, es posible diseñar prácticamente cualquier página web.
- › No obstante, CSS3 define otros selectores más avanzados que permiten simplificar las hojas de estilos y crear auténticas maravillas en el diseño de sitios web.

Selector descendiente (1)

- › Selecciona los elementos que se encuentran dentro de otros elementos. Un elemento es descendiente de otro cuando se encuentra entre las etiquetas de apertura y de cierre del otro elemento.
- › Lo entenderemos mejor con un ejemplo, donde un elemento de párrafo `p` es descendiente de una caja `div`:

```
<div>  
    <p>Este es un elemento descendiente</p>  
</div>
```

Selector descendiente (2)

La manera de escribir una regla en la que se diga, por ejemplo, que todos los párrafos que sean descendientes de una caja tengan textos de color azul sería esta:

```
div p{  
    color:blue;  
}
```

- › En el selector descendiente, un elemento no tiene que ser "*hijo directo*" de otro. La única condición es que un elemento debe estar dentro de otro elemento, sin importar lo profundo que se encuentre.

Selector descendiente (3)

- › El selector del siguiente ejemplo selecciona todos los elementos `` de la página que se encuentren dentro de un elemento `<p>`:

```
p span { color: red; }
```

- › Si el código HTML de la página es el siguiente

```
<p>
...
<span>texto1</span>
...
<a href="">...<span>texto2</span></a>
...
</p>
```

Selector descendiente (4)

El selector `p span` selecciona tanto `texto1` como `texto2`.
El motivo es que en el selector descendente, un elemento no tiene que ser *"hijo directo"* de otro.

Al resto de elementos `` de la página que no están dentro de un elemento `<p>`, no se les aplica la regla CSS anterior.

- › No debe confundirse el selector descendente con la combinación de selectores:

```
/* El estilo se aplica a todos los elementos  
"p", "a" y "span" */  
p, a, span { text-decoration: underline; }  
  
/* El estilo se aplica solo a los elementos  
"span" que se encuentran dentro de "p a" */  
p a span { text-decoration: underline; }
```

Selector de hijos

- › Se trata de un selector similar al selector descendente, pero muy diferente en su funcionamiento.
- › Se utiliza para seleccionar un elemento que es hijo directo de otro elemento y se indica mediante el "signo de mayor que" (>):

```
<p><span>Texto1</span></p>  
<p><a href="#"><span>Texto2</span></a></p>
```

```
p > span { color: blue; }
```

- › En el ejemplo anterior, el selector `p > span` se interpreta como *"cualquier elemento que sea hijo directo de un elemento <p>"*, por lo que el primer elemento cumple la condición del selector. Sin embargo, el segundo elemento no la cumple porque es descendiente pero no es hijo directo de un elemento <p>.

Combinación de selectores (1)

- › CSS permite la combinación de uno o más tipos de selectores para restringir el alcance de las reglas CSS. A continuación se muestran algunos ejemplos habituales de combinación de selectores.

```
.aviso .especial { ... }
```

- › El anterior selector solamente selecciona aquellos elementos con un `class="especial"` que se encuentren dentro de cualquier elemento con un `class="aviso"`.

Combinación de selectores (2)

- › Si se modifica el anterior selector:

```
div.aviso span.especial { ... }
```

- › Ahora, el selector solamente selecciona aquellos elementos de tipo `` con un atributo `class="especial"` que estén dentro de cualquier elemento de tipo `<div>` que tenga un atributo `class="aviso"`.

Selector adyacente(1)

El selector adyacente utiliza el signo + y su sintaxis es:

elemento1 + elemento2 { ... }

La explicación del comportamiento de este selector no es sencilla, ya que selecciona todos los elementos de tipo **elemento2** que cumplan las dos siguientes condiciones:

- › **elemento1** y **elemento2** deben ser hermanos, por lo que su elemento padre debe ser el mismo.
- › **elemento2** debe aparecer inmediatamente después de **elemento1** en el código HTML de la página.

Selector adyacente(2)

```
<body>
  <h1>Titulo1</h1>
  <h2>Subtítulo</h2>
  ...
  <h2>Otro subtítulo</h2>
  ...
</body>
```

```
h1 + h2 {
  color: red
}
```

En el ejemplo anterior los estilos del selector `h1 + h2` se aplican al primer elemento `<h2>` de la página, pero no al segundo `<h2>`, ya que:

- › El elemento padre de `<h1>` es `<body>`, el mismo padre que el de los dos elementos `<h2>`. Así, los dos elementos `<h2>` cumplen la primera condición del selector adyacente.
- › El primer elemento `<h2>` aparece en el código HTML justo después del elemento `<h1>`, por lo que este elemento `<h2>` también cumple la segunda condición del selector adyacente.
- › Por el contrario, el segundo elemento `<h2>` no aparece justo después del elemento `<h1>`, por lo que no cumple la segunda condición del selector adyacente y por tanto no se le aplican los estilos de `h1 + h2`.

Selector de atributos (1)

Los selectores de atributos tienen la cualidad de poder seleccionar elementos en función de una propiedad determinada o de un valor concreto de sus propiedades.

Los dos tipos de selectores de atributos usados habitualmente son los siguientes:

- › **[nombre_atributo]**, selecciona los elementos que tienen establecido el atributo llamado nombre_atributo, independientemente de su valor.
- › **[nombre_atributo=valor]**, selecciona los elementos que tienen establecido un atributo llamado nombre_atributo con un valor igual a valor.

Selector de atributos (2)

- › A continuación se muestran algunos ejemplos de estos tipos de selectores:

```
/* Se muestran de color azul todos los enlaces que tengan  
un atributo "id", independientemente de su valor */  
a[id] { color: blue; }
```

```
/* Se muestran de color gris todos los enlaces que apunten  
al sitio "http://www.udb.edu.sv" */  
a[href="http://www.udb.edu.sv"] { color: gray; }
```

Resumen de selectores CSS

Selector	Afectará a:
*	Cualquier elemento
E	Un elemento (etiqueta HTML) específico, como por ejemplo <code>div</code> o <code>article</code> .
E1 E2	Un elemento (E2) dentro de otro elemento (E1), anidado. Funciona en toda la estructura hereditaria.
E1 > E2	Un elemento (E2) que sea hijo de otro (E1).
E1 + E2	Un elemento (E2) que va detrás y es adyacente a otro elemento (E1)
.clase	Afectará a todo los elementos a los que se le haya asignado esa clase. Ejemplo, <code><div class="caja_lateral"></code>
E.clase	Únicamente al elemento E al que se le haya especificado esa clase.
#ID	Al elemento con el identificador #ID
E#ID	Al elemento E con el identificador ID
E[attr="valor"]	A todos los elementos E cuando el atributo attr sea igual al valor especificado

Selectores CSS3 (1)

CSS3 incluye todos los selectores de CSS2 y añade algunos nuevos selectores, pseudo-clases y pseudo-elementos.

Selector de atributos (nuevas opciones)

CSS 3 añade tres nuevos selectores de atributos:

- › **elemento[atributo^="valor"]**, selecciona todos los elementos que disponen de ese atributo y cuyo valor comienza exactamente por la cadena de texto indicada.
- › **elemento[atributo\$="valor"]**, selecciona todos los elementos que disponen de ese atributo y cuyo valor termina exactamente por la cadena de texto indicada.
- › **elemento[atributo*="valor"]**, selecciona todos los elementos que disponen de ese atributo y cuyo valor contiene la cadena de texto indicada.

Selectores CSS3 (2)

- › De esta forma, se pueden crear reglas CSS tan avanzadas como las siguientes:

```
/* Selecciona todos los enlaces cuyo id comienza  
con la palabra "caja" */  
a[id^="caja"] { ... }
```

```
/* Selecciona todos los enlaces que apuntan a  
una página HTML */  
a[href$=".html"] { ... }
```

```
/* Selecciona todos los títulos h1 cuyo id contenga  
la palabra "capítulo" */  
h1[id*="capítulo"] { ... }
```


Selectores CSS3 (3)

Selector general de hermanos (`elemento1 ~ elemento2`)

Otro de los nuevos selectores de CSS 3 es el "selector general de elementos hermanos", que generaliza el selector adyacente de CSS 2.

Su sintaxis es `elemento1 ~ elemento2` y selecciona el `elemento2` que es hermano de `elemento1` y se encuentra detrás en el código HTML.

En el selector adyacente la condición adicional era que los dos elementos debían estar uno detrás de otro en el código HTML, mientras que ahora la única condición es que uno esté detrás de otro.

Selectores CSS3 (4)

```
<h1>...</h1>
<h2>...</h2>
<p>...</p>
<div>
    <h2>...</h2>
</div>
<h2>...</h2>
```

- › Considerando el código HTML mostrado, se tiene que el selector **h1 + h2** sólo selecciona el primer elemento <h2> de la página, ya que es el único que cumple que es hermano de <h1> y se encuentra justo detrás en el código HTML.
- › Por su parte, el selector **h1 ~ h2** selecciona todos los elementos <h2> de la página salvo el segundo. Aunque el segundo <h2> se encuentra detrás de <h1> en el código HTML, no son elementos hermanos porque no tienen el mismo elemento padre.

Entendiendo la herencia (1)

- › Uno de los conceptos más característicos de CSS es la herencia de los estilos definidos para los elementos. Cuando se establece el valor de alguna propiedad en un elemento, todos sus descendientes heredan inicialmente ese mismo valor.
- › Si se indica por ejemplo un tipo de letra al elemento `<body>` de una página, todos los elementos de la página mostrarán ese tipo de letra, salvo que se indique lo contrario.

Entendiendo la herencia (2)

```
<body>
  <h1>Titular de la pagina</h1>
  <p>Un parrafo de la pagina</p>
</body>
```

```
body {
  font-family: Arial;
  color: black;
}

h1 { font-family: Verdana; }

p { color: red; }
```

- › En el ejemplo anterior, se ha indicado que la etiqueta `<body>` tiene asignado un tipo de letra Arial y un color de letra negro. Así, todos los elementos de la página (salvo que se indique lo contrario) se muestran de color negro y con la fuente Arial.
- › La segunda regla indica que los elementos `<h1>` se muestran con otra tipografía diferente a la heredada. La tercera regla indica que los elementos `<p>` varían su color respecto del color que han heredado. La herencia de estilos no funciona en todas las propiedades CSS, por lo que se debe estudiar cada propiedad de forma individual.

Colisión de estilos (1)

En las hojas de estilos complejas, es habitual que varias reglas CSS se apliquen a un mismo elemento HTML. El problema de estas reglas múltiples es que se pueden dar colisiones como la del siguiente ejemplo:

```
<p>Un parrafo de la pagina</p>
```

```
p { color: red; }  
p { color: blue; }
```

¿De qué color se muestra el párrafo anterior? CSS tiene un mecanismo de resolución de colisiones muy complejo y que tiene en cuenta la importancia de cada regla y lo específico que sea el selector.

Colisión de estilos (2)

A continuación se describe en términos sencillos el método genérico seguido por CSS para resolver las colisiones:

1. Determinar todas las declaraciones que se aplican al elemento para el medio CSS seleccionado.
2. Ordenar las declaraciones según su origen (CSS de navegador, de usuario o de diseñador) y su importancia (palabra clave !important).
3. Ordenar las declaraciones según lo específico que sea el selector. Cuanto más genérico es un selector, menos importancia tienen sus declaraciones.
4. Si después de aplicar las normas anteriores existen dos o más reglas con la misma prioridad, se aplica la que se indicó en último lugar.

Colisión de estilos (3)

En el siguiente ejemplo, la regla CSS que prevalece se decide por lo específico que es cada selector:

```
<p id="especial">Este es un parrafo</p>
```

```
p { color: red; }  
p#especial { color: green; }  
* { color: blue; }
```

- › Al elemento <p> se le aplican las tres declaraciones. Como su origen y su importancia es la misma, decide la especificidad del selector. El selector * es el menos específico, ya que se refiere a *"todos los elementos de la página"*. El selector p es poco específico porque se refiere a *"todos los párrafos de la página"*. Por último, el selector p#especial sólo hace referencia a *"el párrafo de la página cuyo atributo id sea igual a especial"*. Como el selector p#especial es el más específico, su declaración es la que se tiene en cuenta y por tanto el párrafo se muestra de color verde.

PSEUDO-ELEMENTOS



Pseudo-elementos
CSS

¿Qué es un pseudo-elemento?

Es difícil explicar algo tan abstracto como un pseudo-elemento, sin embargo podríamos definirlo como una parte de un elemento que, aunque actúa como éste, no tiene naturaleza de elemento en sí. Un ejemplo, podría ser la primera letra de un párrafo o la primera línea de un texto.

Hay 5 pseudo-elementos: `e:first-letter`, `e:first-line`, `e:first-child`, `e:before` y `e:after`.

En esta clase estudiaremos el uso de los primeros tres pseudo-elementos.

El pseudo-elemento :first-letter

- › El pseudo-elemento `:first-letter` permite seleccionar la primera letra de la primera línea de texto de un elemento.
- › De esta forma, la siguiente regla CSS muestra la primera letra de cada párrafo con el doble del tamaño.

```
p{  
    font-size: 14px;  
}  
  
p:first-letter {  
    font-size: 2em;  
}
```

El pseudo-elemento :first-line

- › El pseudo-elemento :first-line permite seleccionar la primera línea de texto de un elemento.
- › Así, la siguiente regla CSS muestra en mayúsculas la primera línea de cada párrafo:

```
p:first-line {  
    text-transform: uppercase;  
}
```

El pseudo-elemento :first-child

- › La pseudo-clase `:first-child` selecciona el primer elemento hijo de un elemento.

El ejemplo siguiente muestra cómo identificar al primer párrafo dentro de un div sin necesidad de asignar ninguna clase al párrafo.

```
div p:first-child {  
    color: red;  
}
```

PSEUDO-CLASSES



¿Para qué sirven las pseudo-clases?

- › Con las pseudo-clases es posible aplicar efectos especiales a ciertos elementos de un documento, como los vínculos o las imágenes, cuando se realizan algunas acciones específicas con el raton, como pasar sobre una imagen o un enlace o hacer click en ellos.

Las pseudo-clases :link y :visited (1)

Las pseudo-clases `:link` y `:visited` se pueden utilizar para aplicar diferentes estilos a los enlaces de una misma página:

- ❑ La pseudo-clase **`:link`** se aplica a todos los enlaces que todavía no han sido visitados por el usuario.
- ❑ La pseudo-clase **`:visited`** se aplica a todos los enlaces que han sido visitados al menos una vez por el usuario.

Por su propia definición, las pseudo-clases `:link` y `:visited` son mutuamente excluyentes, de forma que un mismo enlace no puede estar en los dos estados de forma simultánea.

Las pseudo-clases :link y :visited (2)

- › Como los navegadores muestran por defecto los enlaces de color azul y los enlaces visitados de color morado, es habitual modificar los estilos para adaptarlos sus necesidades particulares.

```
a:link {  
    color: red;  
}  
  
a:visited {  
    color: green;  
}
```


Las pseudo-clases :hover y :active (1)

Las pseudo-clases **:hover** y **:active** permiten al diseñador web variar los estilos de un elemento en respuesta a las acciones del usuario. Al contrario que las pseudo-clases :link y :visited que sólo se pueden aplicar a los enlaces, estas pseudo-clases se pueden aplicar a cualquier elemento.

A continuación se indican las acciones del usuario que activan cada pseudo-clase:

- ❑ **:hover**, se activa cuando el usuario pasa el ratón o cualquier otro elemento apuntador por encima de un elemento.
- ❑ **:active**, se activa cuando el usuario activa un elemento, por ejemplo cuando pulsa con el ratón sobre un elemento. El estilo se aplica durante un espacio de tiempo prácticamente imperceptible, ya que sólo dura desde que el usuario pulsa el botón del ratón hasta que lo suelta.

La pseudo-clase :nth-child(numero)

› La sintaxis de esta pseudo-clase es:

elemento:nth-child(numero)

Esta pseudo-clase selecciona el elemento indicado pero con la condición de que sea el hijo enésimo de su padre. Este selector es útil para seleccionar el segundo párrafo de un elemento, el quinto elemento de una lista, etc.

Esta pseudo-clase permite el uso de expresiones complejas para realizar selecciones avanzadas.

La pseudo-clase :nth-child(numero)

- › A continuación se muestran ejemplos de uso de esta pseudo-clase:

```
/* selecciona todos parrafos impares
que estan dentro de un div*/
div p:nth-child(2n+1) { ... }

/* selecciona todos parrafos pares
que estan dentro de un div*/
div p:nth-child(2n) { ... }

/* Las siguientes reglas alternan cuatro
estilos diferentes para los párrafos */
p:nth-child(4n+1) { ... }
p:nth-child(4n+2) { ... }
p:nth-child(4n+3) { ... }
p:nth-child(4n+4) { ... }
```

FORMULARIOS

Formulario de Registro

Nombre

Fecha de
Registro

Email

Password :

Enviar

FORMULARIOS

- › Un formulario es una sección del documento web que además de contenido normal (texto, imágenes, etc) posee unos elementos especiales denominados controles o campos de formulario. Estos son el medio con el que el usuario interacciona con el formulario.

Nombre:	<input type="text"/>
Apellido:	<input type="text"/>
Nacionalidad:	<input type="text" value="Salvadoreño(a)"/> 
Sexo:	<input checked="" type="radio"/> Masculino <input type="radio"/> Femenino

FORMULARIOS

Utilidad de los formularios

- › Los formularios permiten obtener información de parte del usuario de la página web.
- › Esta información se recoge a través de controles que el usuario debe ir modificando, ya sea ingresando información o seleccionando opciones.
- › Esta información es enviada a un programa en el servidor web o a una dirección de correo electrónico.

FORMULARIOS

El elemento FORM

- › El elemento FORM es el que se utiliza para poder incorporar un formulario dentro de una página web.
- › La etiqueta de este elemento contiene tres atributos que determinan el comportamiento del formulario, y también puede incluir otros atributos para identificación.

FORMULARIOS

Atributos el elemento FORM

- › **method:** este atributo especifica el método HTTP que se usará para enviar el conjunto de datos del formulario.
- › Los valores posibles pueden ser get (valor por defecto) o post. Ambos no distinguen entre mayúsculas y minúsculas. Si se utiliza el valor get, los datos se envían a través de la URL.

FORMULARIOS

Atributos el elemento FORM

- › ***action***: indica el programa en el servidor web que procesará los datos del formulario. También puede ser que los datos sean enviados a una dirección de correo electrónico.

FORMULARIOS

Atributos el elemento FORM

- › ***enctype***: Especifica el tipo de contenido que será enviado al servidor cuando el valor del atributo method sea post. *El valor por defecto para este atributo es application/x-www-form-urlencoded. Otro valor posible es multipart/form-data, que debería usarse en combinación con elementos INPUT cuyo tipo sea file. Si el valor del atributo action es una dirección de correo electrónico es conveniente establecer el valor de **enctype** como text/plain, para una correcta recepción del mensaje enviado.*

FORMULARIOS

```
<FORM action="http://algunsitio.com/prog/usuarioNuevo" method="post" enctype="">
<P>
<LABEL for="nombre">Nombre:</LABEL> <INPUT type="text" id="nombre"><BR>
<LABEL for="apellido">Apellido:</LABEL> <INPUT type="text" id="apellido"><BR>
<LABEL for="email">email:</LABEL>
<INPUT type="text" id="email"><BR>
<FIELDSET>
<INPUT type="radio" name="sexo" value="Varón"> Varón<BR>
<INPUT type="radio" name="sexo" value="Mujer"> Mujer<BR>
</FIELDSET>
<INPUT type="submit" value="Enviar">
<INPUT type="reset">
</P>
</FORM>
```

FORMULARIOS

Nombre:

Apellido:

email:

☐ Varón

☐ Mujer

Enviar

Reset

FORMULARIOS

π

- › Tipos de controles
- › En un formulario podemos encontrar diferentes tipos de controles. Entre los que se pueden mencionar:
 - Cuadros o cajas de texto
 - Cuadros o cajas de contraseñas
 - Botones de comando
 - Botones de opción
 - Casillas de verificación
 - Menús desplegables
 - Cuadros de lista (con selección única y múltiple)

FORMULARIOS

π

- › Tipos de controles
 - › Áreas de texto
 - › Control de selección de archivos
 - › Controles ocultos
 - › Controles contenedores o de agrupamiento.

FORMULARIOS

π

Elementos para la creación de controles

- › La mayor parte de los controles de formulario se crean con el elemento **INPUT** (cuadros de texto, de contraseñas, botones de opción, casillas de verificación, botones de comando, controles ocultos, etc). Lo que determina el tipo de control para estos controles es el atributo *type de este elemento*.
- › Existe también, el elemento **SELECT**, que permite crear los menús desplegables y los cuadros de lista. Este elemento se utiliza en conjunto con el elemento **OPTION**, que se utiliza para crear las entradas del menú o del cuadro de lista.

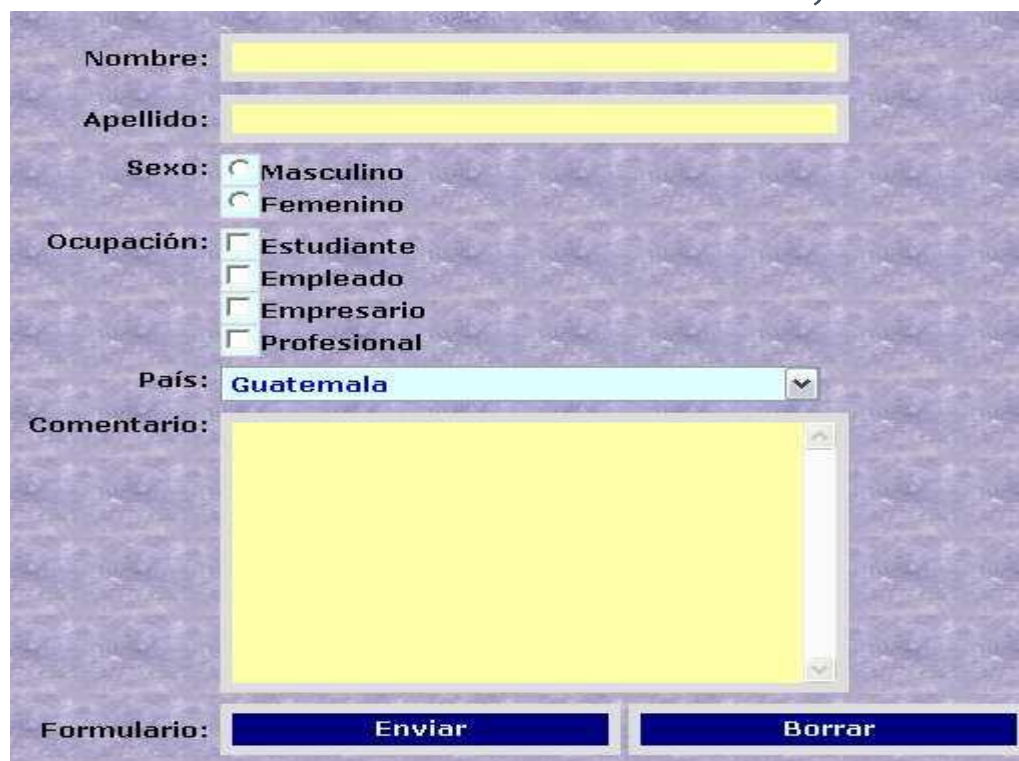
FORMULARIOS

- › Por ejemplo, el elemento **TEXTAREA**, permite crear un control de tipo área de texto. El elemento **FIELDSET** permite crear un control que agrupará otros controles de formulario.
- › Por último, están otros elementos como **LABEL** y **LEGEND**.

FORMULARIOS

- Elemento INPUT

El elemento INPUT permite crear una gran variedad de controles de formulario. Entre ellos el cuadro de texto, el botón de comando, la casilla de verificación, el botón de opción, el control de selección de archivo, controles ocultos, etc.



A screenshot of a web form with a purple background. The form contains the following elements:

- Nombre:** A single-line text input field.
- Apellido:** A single-line text input field.
- Sexo:** Two radio buttons labeled "Masculino" and "Femenino".
- Ocupación:** Four checkboxes labeled "Estudiante", "Empleado", "Empresario", and "Profesional".
- País:** A dropdown menu with "Guatemala" selected.
- Comentario:** A multi-line text area.
- Formulario:** Two buttons at the bottom labeled "Enviar" and "Borrar".

FORMULARIOS

- › El tipo de control que se desea crear queda determinado por el valor del atributo **type** en el elemento **OPTION**. Los valores posibles son muchos y se resumen en la siguiente tabla:

FORMULARIOS

type	control	apariencia
text	Cuadro de texto	<input type="text"/>
password	Cuadro de contraseña	Contraseña: <input type="password"/>
checkbox	Casilla de verificación	<input type="checkbox"/> Macintosh <input type="checkbox"/> PC
radio	Botón de opción	Sexo: <input checked="" type="radio"/> Masculino <input type="radio"/> Femenino
submit	Botón de envío	<input type="submit" value="Enviar"/>
reset	Botón restablecer	<input type="reset" value="Reset"/>
button	Botón pulsador	<input type="button" value="pulsame"/>
image	Botón de envío gráfico	<input type="image" value="pulsame"/> pulsa aquí
file	Control de selección de archivo	<input type="file"/> <input type="button" value="Browse..."/>
hidden	Control oculto	<input type="hidden"/>

FORMULARIOS

Existen muchos atributos más del elemento INPUT, que varían de acuerdo al valor asignado al atributo *type*.

Por ejemplo, si se asigna `type="text"`, se pueden utilizar los siguientes atributos:

name y id : Ambos son identificadores únicos para el cuadro de texto. El primero, especifica un nombre para el control y el segundo, un identificador. Por razones de compatibilidad se sigue usando el name.

value: permite establecer un valor inicial para el control. En el caso de un cuadro de texto, el texto por defecto que aparecerá en el control.

FORMULARIOS

- › **size:** El tamaño con que se visualizará el control medido en número de caracteres.
- › **maxlength:** define el número máximo de caracteres que se pueden ingresar en el control de cuadro de texto. Por defecto, se pueden ingresar infinitos caracteres.
- › **disabled:** hace que el control aparezca deshabilitado en el contexto.

FORMULARIOS

- › **readonly:** hace que el contenido del control sea de sólo lectura; es decir, no se puede modificar.
- › **alt:** permite colocar un texto descriptivo que se mostrará en forma de tool tip text cuando el usuario coloque el puntero del ratón encima del control.
- › **accesskey:** permite establecer una tecla de acceso rápido al control.
- › **tabindex:** permite establecer la posición que le corresponde al control en el orden de tabulación

FORMULARIOS

- Elemento SELECT

El elemento SELECT permite crear controles de tipo menú desplegable y cuadros de lista. En este caso lo que determina el tipo específico de control es el valor del atributo `size`.

Si el valor de `size` es “1”, el control será un menú desplegable. Si es mayor que uno será un cuadro de lista. Los cuadros de lista pueden permitir selección múltiple si se utiliza adicionalmente el atributo `multiple`. Si no se utiliza solamente se puede seleccionar una opción dentro del cuadro de lista.

FORMULARIOS

Los atributos del elemento SELECT se muestran a

continuación:

- › **name y id:** realizan la misma función que para el elemento INPUT, sirven para asignar un nombre único al control y para establecer un identificador único al mismo, respectivamente.
- › **size:** permite indicar el número de filas que tendrá el cuadro de lista (valores mayores que 1). Si el valor es uno el control mostrado será un menú desplegable.

FORMULARIOS

- › **multiple:** este atributo booleano, si es utilizado, permite que se puedan seleccionar varias opciones o todas en el cuadro de lista. Solamente tiene sentido utilizarlo cuando el control sea un cuadro de lista. Cuando no se utiliza este atributo, el cuadro de lista solamente permite seleccionar una opción. Para seleccionar múltiples opciones puede presionar la tecla <CTRL> y sin soltarla hacer clic con el ratón encima de la opción deseada.

FORMULARIOS

› Elemento OPTION

El elemento SELECT no se utiliza solo. Para producir un menú desplegable o un cuadro de lista deben utilizarse, dentro del elemento SELECT, uno o varios elementos OPTION. Cada uno de los elementos OPTION produce una nueva entrada para el menú desplegable o cuadro de lista.

FORMULARIOS

Los atributos del elemento **OPTION** se describen a

continuación:

- › **value:** permite establecer el valor inicial para el control. Si no se establece se utiliza el contenido del elemento **OPTION**.
- › **label:** Permite especificar un rótulo más corto que el contenido del elemento **OPTION**.
- › **selected:** este atributo booleano hace que la opción actual aparezca preseleccionada.

FORMULARIOS

- › Elemento **OPTGROUP**
- › Este elemento permite agrupar opciones de forma lógica. Pueden agruparse opciones si están divididas en categorías.
- › Todos los elementos **OPTGROUP** deben especificarse directamente dentro de un elemento **SELECT**.
- › El único atributo del elemento **OPTGROUP** es label, que especifica un rótulo para el grupo de opciones.

FORMULARIOS

› Elemento TEXTAREA

Este elemento permite crear un control de entrada de texto de varias líneas. El contenido que se coloca entre las etiquetas `<TEXTAREA> ... </TEXTAREA>` debe ser utilizado como valor inicial para el control y debe aparecer dentro del control cuando se muestre en el navegador.



FORMULARIOS

Los atributos del elemento **TEXTAREA** se describen a

continuación:

- › ***name y id***: Asigna un nombre único al control y establece un identificador único para el control, respectivamente.
- › ***rows***: especificar el número de líneas de texto visibles. Los usuarios podrán introducir más líneas que las visibles, para ello el navegador debe proporcionar un mecanismo para desplazarse a las líneas que no sean visibles en un momento determinado.

FORMULARIOS

cols: establece el ancho visible para el control, medido en número de caracteres. También se pueden ingresar líneas de mayor longitud que la visible. El navegador debe hacer aparecer una barra de desplazamiento para poder desplazarse.

FORMULARIOS

› Elementos FIELDSET y LEGEND

Este elemento permite agrupar varios controles y etiquetas con el objeto de dar más estructura a los formularios. El elemento LEGEND permite agregar un rótulo al grupo de campos de formularios y a sus etiquetas.

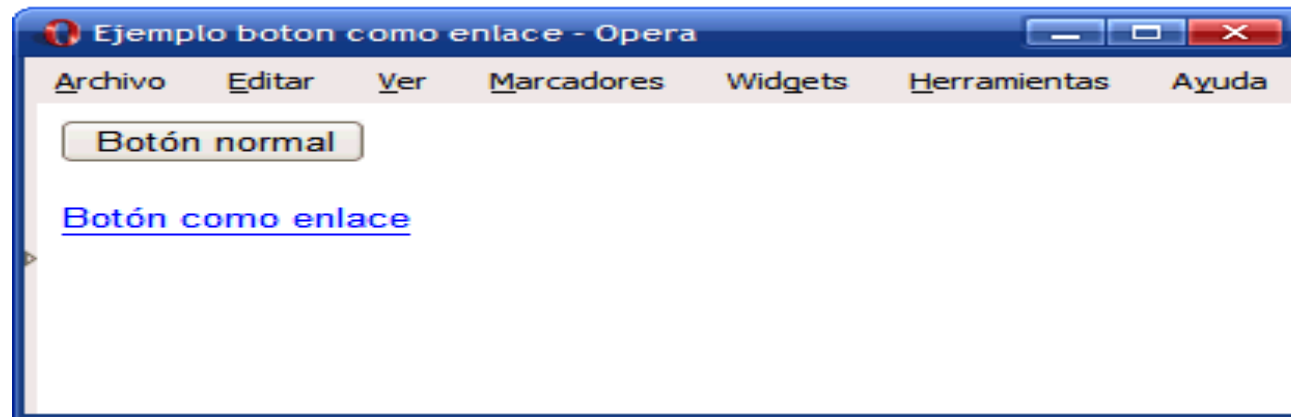
Datos personales:	
Nombre:	<input type="text"/>
Apellido:	<input type="text"/>
Edad:	<input type="text"/>
Sexo:	<input checked="" type="radio"/> Masculino <input type="radio"/> Femenino
Nacionalidad:	El Salvador <input type="button" value="v"/>

FORMULARIOS

› Estilos básicos con CSS

Mostrar un botón como un enlace

Los botones de formulario también se pueden modificar para que parezcan enlaces.



FORMULARIOS

Mostrar un botón como un enlace

- › Las reglas CSS del ejemplo anterior son las siguientes:

```
.enlace {  
border: 0;  
padding: 0;  
background-color: transparent;  
color: blue;  
border-bottom: 1px solid blue;  
}
```

```
<input type="button" value="Botón normal" />
```

```
<input class="enlace" type="button" value="Botón  
como enlace" />
```

FORMULARIOS

Mejoras en los campos de texto

Por defecto, los campos de texto de los formularios no incluyen ningún espacio de relleno, por lo que el texto introducido por el usuario aparece *pegado a los bordes del* cuadro de texto.

Añadiendo un pequeño padding a cada elemento `<input>`, se mejora notablemente el aspecto del formulario:

FORMULARIOS

π

The screenshot shows a web browser window with the title "Ejemplo padding en input". The browser has a menu bar with "Archivo", "Edición", "Ver", "Marcadores", "Widgets", "Titulares", "Herramientas", and "Ayuda". The window displays two forms side-by-side for comparison.

Formulario sin padding en los input

Nombre

Contraseña

Formulario con padding en los input

Nombre

Contraseña

FORMULARIOS

π

Mejoras en los campos de texto

- › La regla CSS necesaria para mejorar el formulario es muy sencilla:

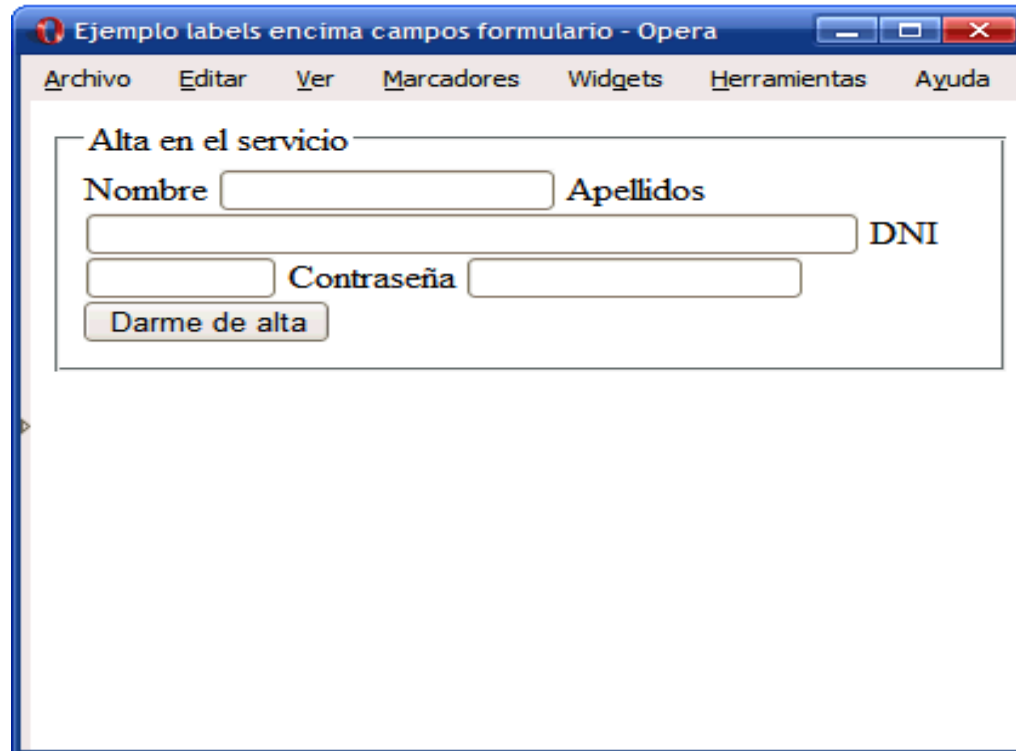
```
form.elegante input {  
padding: .2em;  
}
```

FORMULARIOS

π

Labels alineadas y formateadas

Los elementos `<input>` y `<label>` de los formularios son elementos en línea, por lo que el aspecto que muestran los formularios por defecto, es similar al de la siguiente imagen:



The image shows a screenshot of a web browser window titled "Ejemplo labels encima campos formulario - Opera". The browser's menu bar includes "Archivo", "Editar", "Ver", "Marcadores", "Widgets", "Herramientas", and "Ayuda". The main content area displays a form titled "Alta en el servicio". The form contains the following elements:

- A label "Nombre" followed by an input field.
- A label "Apellidos" followed by an input field.
- A label "DNI" followed by an input field.
- A label "Contraseña" followed by an input field.
- A button labeled "Dar de alta".

The labels are positioned above their respective input fields, and the button is located below the password field.

FORMULARIOS

- El código HTML del ejemplo anterior es el siguiente:

```
<form>
```

```
<fieldset>
```

```
<legend>Alta en el servicio</legend>
```

```
<label for="nombre">Nombre</label>
```

```
<input type="text" id="nombre" />
```

```
<label for="apellidos">Apellidos</label>
```

```
<input type="text" id="apellidos" size="50" />
```

```
<label for="dni">DNK</label>
```

```
<input type="text" id="dni" size="10" maxlength="9" />
```

```
<label for="contrasena">Contraseña</label>
```

```
<input type="password" id="contrasena" />
```

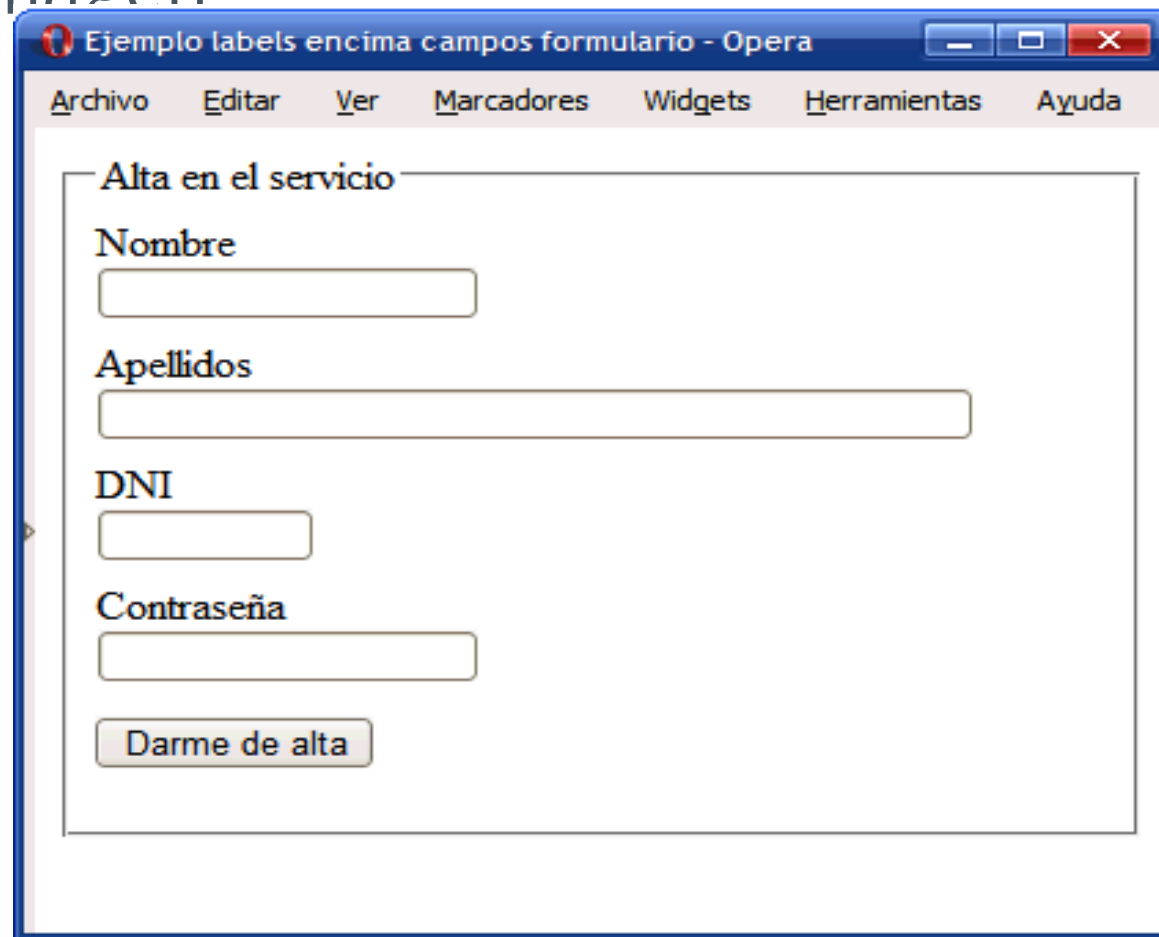
```
<input class="btn" type="submit" value="Dar de alta" />
```

```
</fieldset>
```

```
</form>
```

FORMULARIOS

- › Aprovechando los elementos `<label>`, se pueden aplicar unos estilos CSS sencillos que permitan mostrar el formulario con el aspecto de la siguiente imagen.



The image shows a screenshot of a web browser window titled "Ejemplo labels encima campos formulario - Opera". The browser's menu bar includes "Archivo", "Editar", "Ver", "Marcadores", "Widgets", "Herramientas", and "Ayuda". The main content area displays a registration form titled "Alta en el servicio". The form contains the following elements:

- Nombre**: A text input field.
- Apellidos**: A text input field.
- DNI**: A text input field.
- Contraseña**: A text input field.
- Darme de alta**: A button.

FORMULARIOS

- › En primer lugar, se muestran los elementos `<label>` como elementos de bloque, para que añadan una separación para cada campo del formulario. Además, se añade un margen superior para no mostrar juntas todas las filas del formulario:

```
label {  
  display: block;  
  margin: .5em 0 0 0;  
}
```

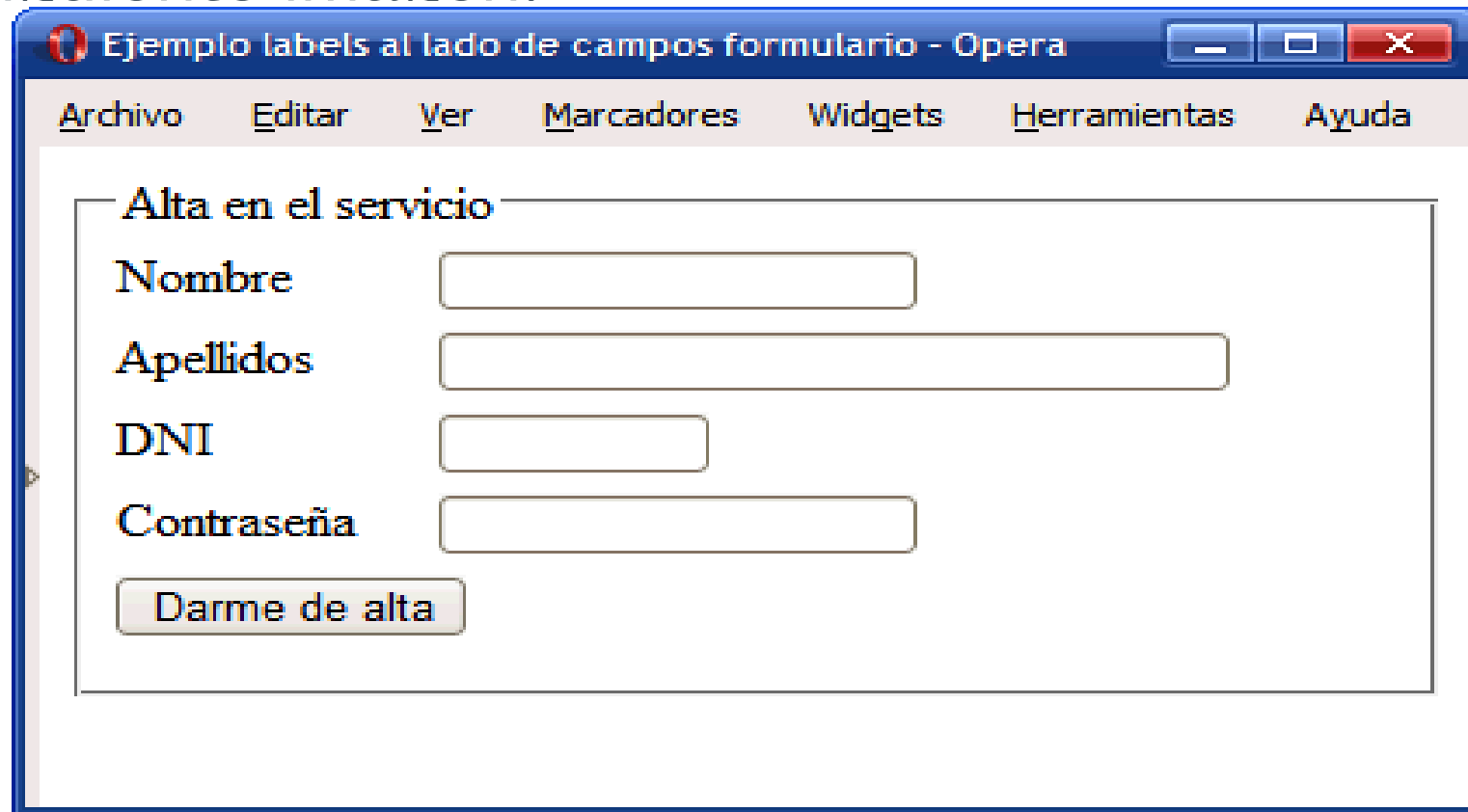
FORMULARIOS

- › El botón del formulario también se muestra como un elemento de bloque y se le añade un margen para darle el aspecto final deseado:

```
.btn {  
  display: block;  
  margin: 1em 0;  
}
```

FORMULARIOS

- › En ocasiones, es más útil mostrar todos los campos del formulario con su <label> alineada a la izquierda y el campo del formulario a la derecha de cada <label>, como muestra la siguiente imagen:



The image shows a screenshot of a web browser window titled "Ejemplo labels al lado de campos formulario - Opera". The browser's menu bar includes "Archivo", "Editar", "Ver", "Marcadores", "Widgets", "Herramientas", and "Ayuda". The main content area displays a registration form titled "Alta en el servicio". The form contains four labels ("Nombre", "Apellidos", "DNI", "Contraseña") aligned to the left, each followed by a corresponding text input field. Below these fields is a button labeled "Dar me de alta".

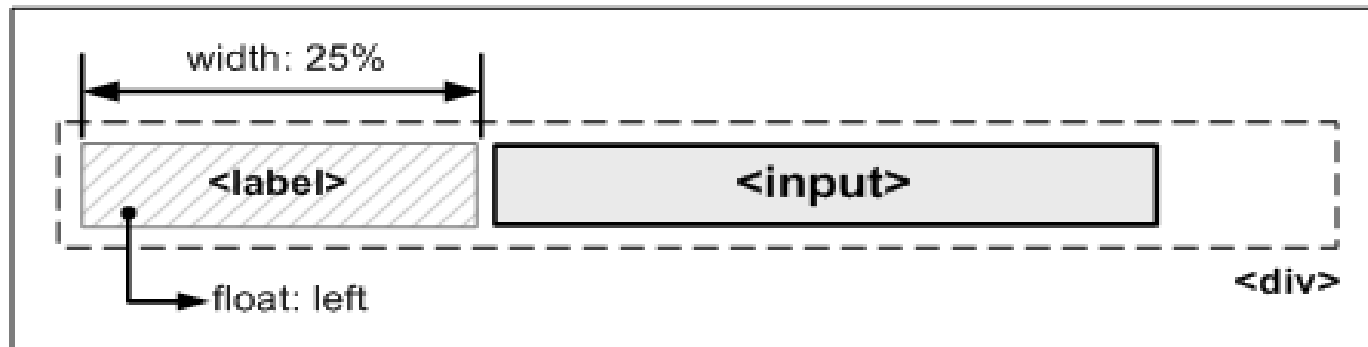
Alta en el servicio	
Nombre	<input type="text"/>
Apellidos	<input type="text"/>
DNI	<input type="text"/>
Contraseña	<input type="password"/>
<input type="button" value="Dar me de alta"/>	

FORMULARIOS

π

- › Para mostrar un formulario tal y como aparece en la imagen anterior no es necesario crear una tabla y controlar la anchura de sus columnas para conseguir una alineación perfecta. Sin embargo, sí que es necesario añadir un nuevo elemento (por ejemplo un `<div>`) que encierre a cada uno de los campos del formulario (`<label>` y `<input>`). El

esquema de la solución propuesta es el siguiente:



FORMULARIOS

- Por tanto, en el código HTML del formulario anterior se añaden los elementos `<div>`:

`<form>`

`<fieldset>`

`<legend>Alta en el servicio</legend>`

`<div>`

`<label for="nombre">Nombre</label>`

`<input type="text" id="nombre" />`

`</div>`

`<div>`

`<label for="apellidos">Apellidos</label>`

`<input type="text" id="apellidos" size="35" />`

`</div>`

`...`

`</fieldset>`

`</form>`

FORMULARIOS

π

- › Y en el código CSS se añaden las reglas necesarias para alinear los campos del formulario:

```
div {  
margin: .4em 0;  
}  
  
div label {  
width: 25%;  
float: left;  
}
```

FORMULARIOS

Estilos avanzados

› Formulario en varias columnas

Los formularios complejos con decenas de campos pueden ocupar mucho espacio en la ventana del navegador. Además del uso de pestañas para agrupar los campos relacionados en un formulario, también es posible mostrar el formulario a dos columnas, para aprovechar mejor el espacio.

The screenshot shows a web browser window with the title "Ejemplo formulario 2 columnas - Opera". The browser's menu bar includes "Archivo", "Editar", "Ver", "Marcadores", "Widgets", "Herramientas", and "Ayuda". The form is divided into two columns by a vertical line. The left column is titled "Alta en el servicio" and contains four input fields: "Nombre", "Apellidos", "DNI", and "Contraseña". The right column is titled "Datos de contacto" and contains three input fields: "Telefono", "Email", and "Dirección", followed by a "Código Postal" field. At the bottom of the form, centered across both columns, is a button labeled "Dar de alta".

FORMULARIOS

π

- › La solución consiste en aplicar la siguiente regla CSS a los `<fieldset>` del formulario:

```
form fieldset {  
  float: left;  
  width: 48%;  
}
```

```
<form>
```

```
<fieldset>
```

```
</fieldset>
```

```
...
```

```
</form>
```


FORMULARIOS

Si se quiere mostrar el formulario con más de dos columnas, se aplica la misma regla pero modificando el valor de la propiedad `width` de cada `<fieldset>`. Si el formulario es muy complejo, puede ser útil agrupar los `<fieldset>` de cada fila mediante elementos `<div>`.

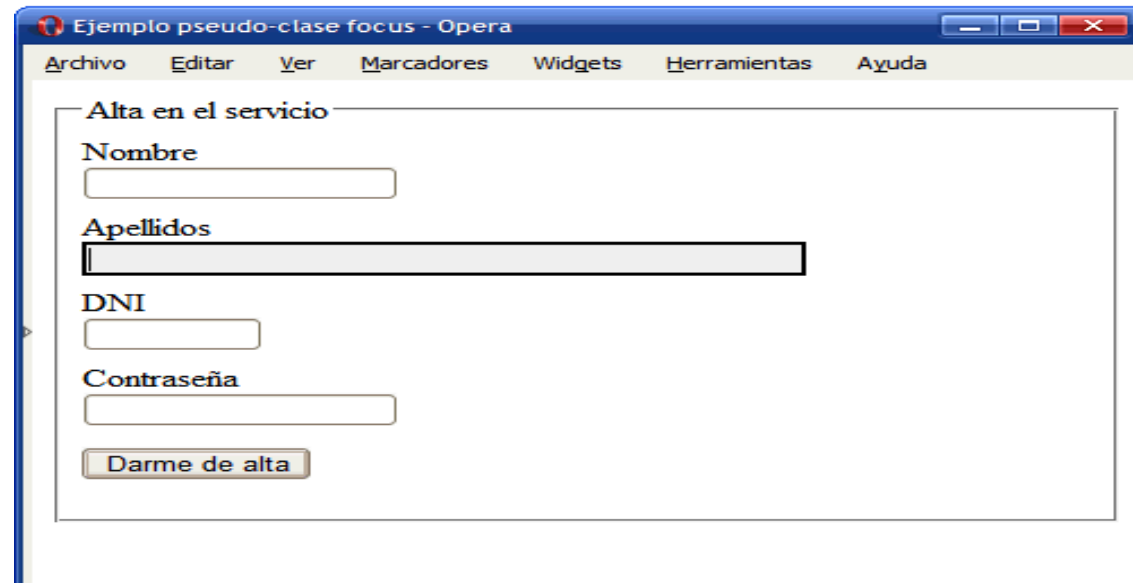
FORMULARIOS

π

Resaltar el campo seleccionado

- Una de las mejoras más útiles para los formularios HTML consiste en resaltar de alguna forma especial el campo en el que el usuario está introduciendo datos. Para ello, CSS

define la pseudo-clase: focus, que permite aplicar estilos especiales al elemento que en ese momento tiene el foco o atención del usuario.



The screenshot shows a web browser window titled "Ejemplo pseudo-clase focus - Opera". The browser's menu bar includes "Archivo", "Editar", "Ver", "Marcadores", "Widgets", "Herramientas", and "Ayuda". The main content area displays a form titled "Alta en el servicio". The form contains the following fields and elements:

- Nombre**: A text input field.
- Apellidos**: A text input field that is currently highlighted with a thick black border, indicating it has the focus.
- DNI**: A text input field.
- Contraseña**: A text input field.
- Darme de alta**: A button at the bottom of the form.

FORMULARIOS

Añadiendo la pseudo-clase :focus después del selector normal, el navegador se encarga de aplicar esos estilos cuando el usuario activa el elemento:

```
input:focus {  
border: 2px solid #000;  
background: #F3F3F3;  
}
```