

# Uso de estilos



---

# Introducción a CSS

- Permite separar en el desarrollo de un sitio web diseño de contenido.
  - Mantenimiento más eficiente de sitios web.
  - Definición de CSS:
    - Mediante atributo style de HTML
    - Etiqueta <style> dentro de <head>
    - En un fichero externo .css
-

# Definición de CSS

- Varias formas de definir estilos:
  - Usando el atributo *style* dentro de una etiqueta.
    - Ej: <p style="background: black;">párrafo</p>
    - Presenta alta prioridad frente a otras reglas. Sin embargo es una desventaja porque
      - Imposibilita la reutilización
      - Dificulta el mantenimiento CSS.
  - Etiqueta *style* dentro de la cabecera. Esta etiqueta tiene varios atributos opcionales:
    - Type (requerido): para indicar que se aplica un estilo CSS
    - Media: para indicar el tipo de dispositivo.
    - Title: nombre que se le da al estilo

# Hojas de estilo en fichero externo

- La ventaja de utilizar hojas de estilo externas es que se pueden reutilizar en varios documentos.
- Puede ser enlazada a un documento mediante la etiqueta `<link>` que se coloca en la cabecera.

Ej: `<link rel=stylesheet href="estilo.css" type="text/css">`

- El fichero css debe contener únicamente reglas de estilo.

---

# Tipos de selectores.

- Estilos basados en:
  - Etiquetas
  - Clases
  - Identificadores

# Selectores: etiquetas

- Selectores basados en etiquetas:
  - Usando las propias etiquetas HTML como selectores.
    - Selector { atributo: valor}
    - El atributo hace referencia a la característica que se quiere modificar de la etiqueta (p.e. color).
    - El valor hace referencia a la instancia del atributo.
    - Por ejemplo: h1 {color:blue}

# Selectores: etiquetas

- Selectores basados en etiquetas:
  - Se pueden definir atributos para varios selectores y selectores con varios atributos.

Selector1, selector2 {atributo1:valor1; atributo2:valor2}

# Selectores: clases

- **Selectores basados en clases:**
  - Mediante las clases se pueden definir estilos abstractos que no estén asociados a una etiqueta HTML.
  - Permiten aplicar estilos a etiquetas HTML, con el mismo efecto que usando selectores de etiquetas, pero también a cualquier otro elemento de la página.
  - Hay diferentes tipos de clases:
    - ❑ Asociadas directamente a una etiqueta HTML
    - ❑ Genéricas, que se pueden aplicar a cualquier etiqueta.



# Selectores: clases

- Selectores basados en clases:
  - Nombreetiqueta.nombreclase {atributo:valor}
    - Ejemplos:
      - h1.verde {color:green}
      - h1.azul {color:blue}
  - Para indicar que estilo aplicar a cada etiqueta:
    - <h1 class="nombreclase">texto</h1>

# Selectores: clases

- Selectores basados en clases:
  - Las clases más genéricas no se aplican a ninguna etiqueta HTML por lo que en su descripción no se especifica el nombre de ninguna etiqueta.
    - `.nombreclase {atributo:valor}`
    - Por ejemplo: `.verde {color:green}`
  - Una clase así definida se puede aplicar a cualquier elemento de la página.

# Selectores: identificadores

- Selectores basados en identificadores:
  - Los selectores basados en identificadores son muy similares a los basados en clases y con una sintaxis muy parecida.
  - La diferencia es que los identificadores solo se pueden usar en un único elemento
    - Nombreetiqueta#nombreidentificador {atributo:valor}
    - #nombreidentificador {atributo:valor}

---

# Selectores: identificadores

- Selectores basados en identificadores:
  - `#verde {color:green}`
  - `<p id="verde">Párrafo en color verde</p>`

# Selectores: identificadores

- Los identificadores se suelen usar cuando identificadores como *cabecera*, *contenidos* y *piepagina* definen el estilo de tres zonas de una página web.
- No tiene sentido que esos identificadores se repitan en varios elementos.
- Ejemplo:
  - ❑ `<div id="cabecera"></div>`
  - ❑ `<div id="contenido"></div>`
  - ❑ `<div id="piepagina"></div>`

# Agrupación y anidamiento de selectores

- Los selectores se pueden agrupar y anidar para conseguir estilos CSS.
  - Agrupamientos
    - Hace referencia a la manera en la que se pueden escribir las reglas de estilo para conseguir un CSS más claro y fácil de entender.
    - Se puede aplicar el mismo estilo a un conjunto de selectores al mismo tiempo.
    - Selector1, selector2 {atributo1:valor1; atributo2:valor2}

# Agrupación y anidamiento de selectores

- ❑ Los selectores se pueden anidar con el fin de conseguir estilos más concretos y definidos.
- ❑ En CSS se llaman selectores contextuales.
  - Selector anidado común: se usa para crear reglas sobre elementos que están rodeados de otros elementos.
    - ❑ Ej: `h1 i b {color:blue}`
  - Anidamiento de selectores hijos: si lo que se desea es restringir que las etiquetas, además de estar en el mismo contexto, estén seguidas unas de otras
    - ❑ Ej: `h1>b {color:blue}`

# Agrupación y anidamiento de selectores

- ❑ Anidamiento de selectores adyacentes: se usa cuando se quiere aplicar un estilo a un elemento que tiene adyacente a otro elemento en el mismo nivel de anidamiento en HTML.

Ej: `i+b {color:yellow}`

`<i>Nota</i>, esto es una <b>advertencia</b>`

`<b>Leer detenidamente</b>`

- ❑ En este ejemplo solo se aplicará el estilo a la palabra *advertencia*.



# Agrupación y anidamiento de selectores

- ❑ Anidamiento de selectores hermanos: se usa cuando se quiere aplicar un estilo a un elemento que tiene como hermano a otro elemento HTML.

```
div~p  
{ background-color:yellow;}
```

---

# Buenas prácticas al escribir CSS

- Recomendaciones a la hora de escribir CSS para que las modificaciones posteriores sean más fáciles. No están definidas en el estándar W3C, pero ayudan en el desarrollo.
    - ❑ Los selectores se nombran en minúsculas, nunca empezando por caracteres especiales o numéricos.
    - ❑ El nombre de los selectores debe ser específico y claro, para que tenga una mayor capacidad expresiva.
-

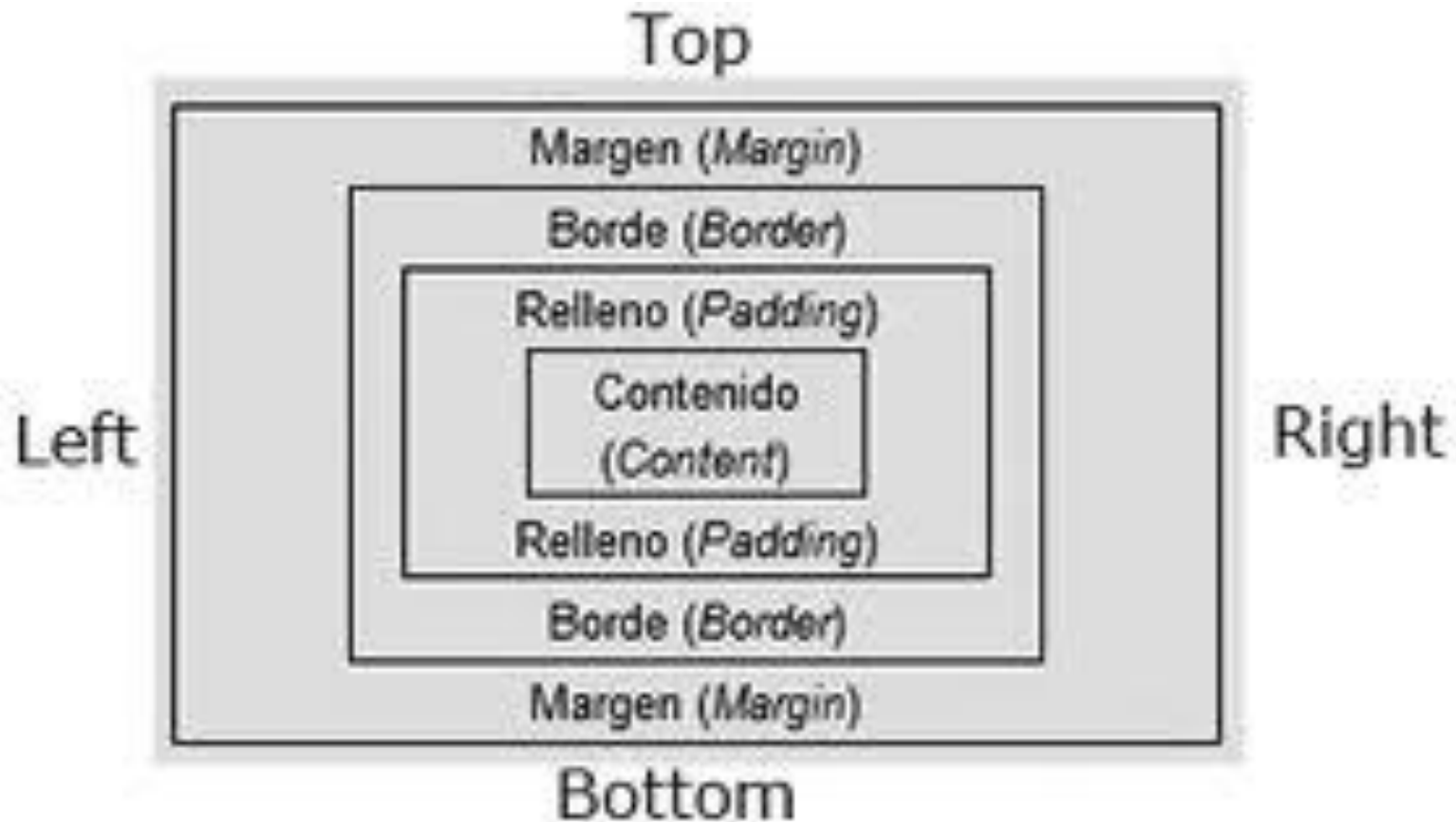
# Buenas prácticas al escribir CSS

- ❑ El nombre de las clases e identificadores no debe describir una característica visual, como color, tamaño o posición. Si especificamos un nombre definiendo un color y cambia el color de la clase también se debería cambiar el nombre del selector.
- ❑ Los nombres deben seguir más una visión semántica que estructural.
- ❑ Comentarios: `/*texto del comentario*/`

# Buenas prácticas al escribir CSS

- ❑ Separa las palabras mediante guiones o mayúsculas.
- ❑ No hacer uso excesivo de clases: utilizar en su lugar selectores contextuales o anidados.
- ❑ Agrupar estilos según selector siempre que sea posible:
  - table
  - table.empleados
  - Etc
- ❑ Al principio de un CSS es aconsejable definir los selectores de etiquetas.

# ATRIBUTOS. MODELO DE CAJAS



# Atributos. Modelo de cajas.

- Unidades de medida:

- ❑ Pulgadas
- ❑ Centímetros
- ❑ Milímetros
- ❑ Puntos
- ❑ Picas

- Atributos de posición:

- ❑ Top, left, right, bottom

# Atributos. Modelo de cajas.

- Atributos margin:
  - Margin-left, margin-right, margin-top, margin-bottom
- Atributos padding: se puede traducir como relleno. Indica la distancia entre el borde y los elementos que se encuentran en el interior.
  - Padding-top, padding-right, padding-bottom, padding-left

# Atributos. Modelo de cajas.

- Atributos border: definen el estilo y color del borde de la caja [https://www.w3schools.com/css/css\\_border.asp](https://www.w3schools.com/css/css_border.asp)
  - Border-top, border-bottom, border-right, border-left
  - Algunos de las palabras claves son:
    - None
    - Dotted
    - Dashed
    - Solid
    - Double
    - Groove
    - Ridge
    - Inset
    - outset

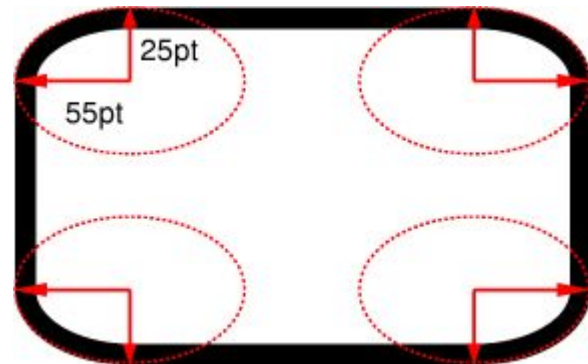


# Atributos. Modelo de cajas.

## ■ Atributos border:

- ❑ Border-radius: se usa para realizar bordes redondeados.
- ❑ Ejemplo:

```
div {  
  border: 2px solid;  
  border-radius: 25px;  
}
```



# ELEMENTOS: textos

- Una vez vistos los atributos asociados a las clases estos son los atributos relacionados con la apariencia de textos, listas, tablas, enlaces e imágenes.
- **Fuentes:** [https://www.w3schools.com/css/css\\_font.asp](https://www.w3schools.com/css/css_font.asp)
  - color: nombres en inglés y valores RGB.
  - font-size: unidades | xx-small | x-small | small | medium | large | x-large | xx-large
  - font-family. Serif | sans-serif | cursive | fantasy | monospace. Indica la tipografía del texto.
  - font-weight: normal | bold | bolder | lighter | 100 | 200 | ... | 900. Para definir la anchura de los caracteres (efecto negrita) Normal=400, bold=700
  - font-style: normal | italic | oblique. Estilo de la fuente. Oblique es similar a italic.

# ELEMENTOS: textos

## □ **Párrafos:**

- line-height: normal | unidades. El alto de una línea y por tanto el espaciado entre líneas.
- text-decoration: none | underline | overline | line-through. Para establecer la decoración de un texto, si está subrayado, sobre-rayado o tachado.
- text-align: alineación del texto.
- text-indent: Establece tabulación primera línea.
- text-transform: capitalize | uppercase | lowercase | none | text-transform | none. Permite transformar el texto haciendo que tenga la primera letra en mayúsculas, de todas las palabras, todo en mayúsculas o minúsculas.

# ELEMENTOS: fondo y tablas

- ❑ **Fondo:** [https://www.w3schools.com/css/css\\_background.asp](https://www.w3schools.com/css/css_background.asp)
  - background-color.
  - background-image.
  - background-repeat
- ❑ **Tablas:**
  - caption-side: top | bottom. Posición del título.
  - table-layout: auto | fixed.
  - border-collapse: collapse | separate. Selección del modelo de los bordes.
  - border-spacing: unidades. Espaciado entre los bordes de celdas adyacentes.
  - empty-cells: show | hide. Visibilidad de los bordes de celda sin contenido.

# ELEMENTOS: visibilidad

## □ **Visibilidad:**

- overflow: visible | hidden | scroll | auto. Comportamiento del contenido si se desborda en la caja.
- clip: rect (top, right, bottom, left) | auto. Especifica la región visible del elemento mediante las dimensiones de un rectángulo que hace de ventana de visualización.
- visibility: visible | hidden | collapse. Visibilidad de las cajas.
- display: muestra una caja con diferentes estilos. El más común es none, que se diferencia de **visibility: hidden** en que en este caso las cajas de alrededor se reorganizan cuando se oculta.

# ELEMENTOS: listas

## ❑ **Listas:**

- list-style-type: disc | circle | square.... Estilo aplicable a las listas.
- list-style-image: url(<http://...>) | none. Imagen aplicable a los elementos de las listas.
- list-style-position: inside | outside

# Pseudo-clases y pseudo-elementos

# CSS pseudo-clases

Definen diferentes estados de un elemento.  
Por ejemplo:

## ❑ Enlaces

- Normales: a:link {atributos}
- Visitados: a:visited {atributos}
- Activos: a:active {atributos}. Los enlaces están activos en el preciso momento en que se pulsa sobre ellos.
- Hover: a:hover {atributos}. Cuando el ratón está encima.



# CSS pseudo-classes

```
/* unvisited link */  
a:link {  
    color: #FF0000;  
}
```

```
/* visited link */  
a:visited {  
    color: #00FF00;  
}
```

```
* mouse over link */  
:hover {  
    color: #FF00FF;
```

```
* selected link */  
:active {  
    color: #0000FF;
```

# CSS pseudo-elementos

Definen diferentes estilos para parte de un elemento

```
p::first-line {  
  color: #ff0000;  
  font-variant: small-caps;  
}
```

```
p::first-letter {  
  color: #ff0000;  
  font-size: xx-large;  
}
```

# CSS pseudo-elementos

```
h1::before {  
    content: url(smiley.gif);  
}
```

---

```
h1::after {  
    content: url(smiley.gif);  
}
```

---

# Ejemplo (HTML) - Tooltip

```
<p>Here we have some <span tabindex="0" data-descr="collection of words and punctuation">text</span> with a few <span tabindex="0" data-descr="small popups that appear when hovering">tooltips</span>. </p>
```

[https://mdn.mozillademos.org/files/4591/css-only\\_tooltips.html](https://mdn.mozillademos.org/files/4591/css-only_tooltips.html)

# Ejemplo (CSS) - Tooltip

```
span[data-descr] {  
    position: relative;  
    text-decoration: underline;  
    color: #00F;  
    cursor: help;  
}
```

# Ejemplo (CSS) - Tooltip

```
span[data-descr]:hover::after,  
span[data-descr]:focus::after {  
  content: attr(data-descr);  
  position: absolute;  
  left: 0;  
  top: 24px;  
  min-width: 200px;  
  border: 1px #aaaaaa solid;  
  border-radius: 10px;  
  background-color: #ffffcc;  
  padding: 12px;  
  color: #000000;  
  font-size: 14px;  
  z-index: 1;
```

# Posicionamiento

---

# Tipos de elementos (según display)

- El atributo *display* puede tomar los siguientes valores:
  - ❑ **block**: elemento de bloque, ocupa la totalidad de la línea
  - ❑ **inline**: elemento de línea, ocupa el contenido del elemento
  - ❑ **inline-block**: elemento de línea pero admite atributos width y height.

Los párrafos son elementos de bloque.

Los enlaces son elementos en línea

Dentro de un párrafo, los enlaces siguen siendo elementos en línea.



# Posicionamiento (position)

- Position puede tener como valores:
  - ❑ **Static:** valor predeterminado. No provoca ningún posicionamiento especial de los elementos y por tanto, los atributos *top*, *left*, *right* y *bottom* no se tendrán en cuenta.
  - ❑ **Relative:** se ve influenciado por los elementos anteriores pero *top* y *left* definen la distancia respecto al último elemento.
  - ❑ **Absolute:** permite posicionar cajas de manera absoluta de manera definida mediante los valores *top*, *left*, *bottom* y *right*. Los elementos no se ven afectados por el lugar de otros elementos. Si la posición absoluta es respecto al contenedor, este debe ser relative.
  - ❑ **Fixed:** Posiciona el elemento según posicionamiento absoluto, pero su posición final será siempre fija. Admiten valores *top* y *left*.
  - ❑ **Sticky:** Es una mezcla de relative y fixed manteniendose fijo cuando el scroll baja.

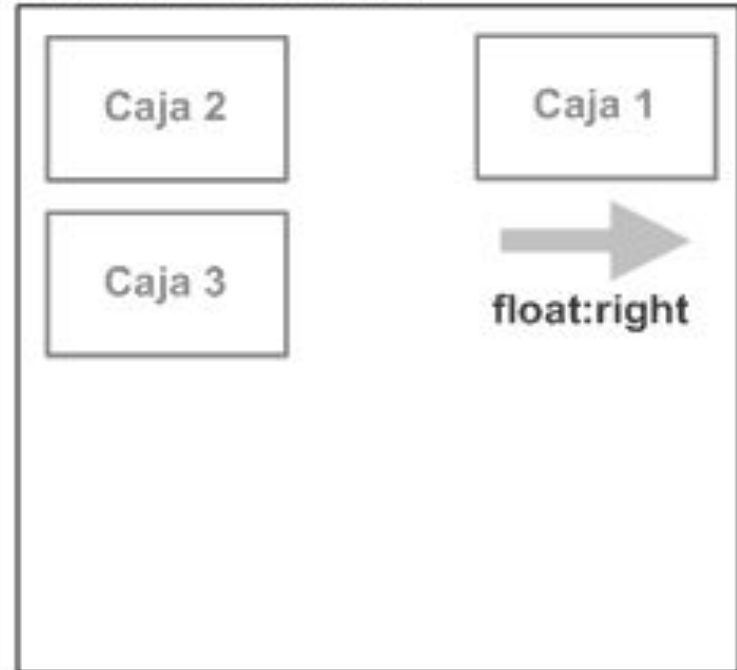
# Posicionamiento flotante

Elemento contenedor



Posicionamiento normal

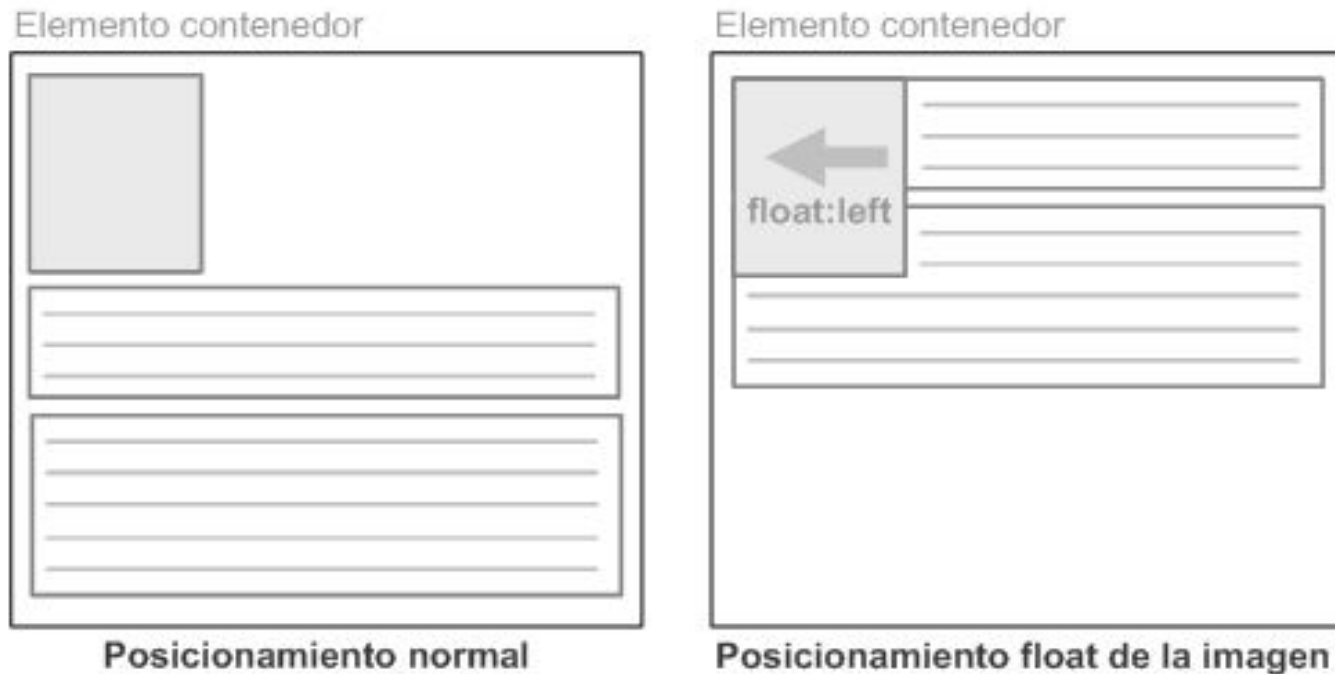
Elemento contenedor



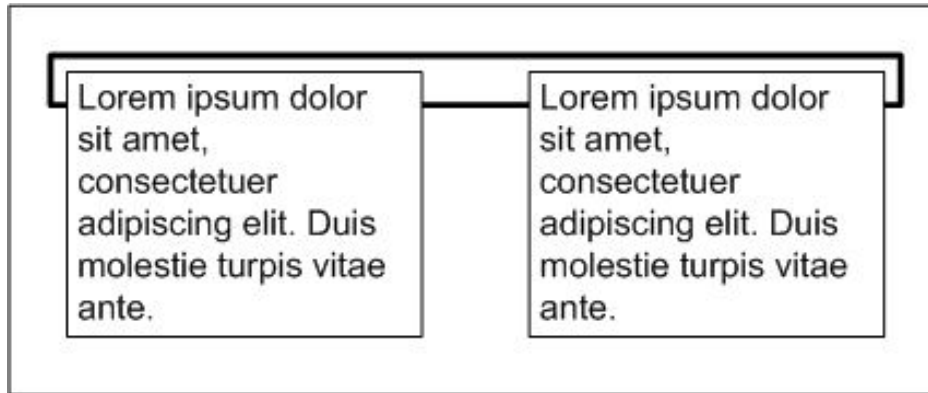
Posicionamiento float de la caja 1

# Posicionamiento flotante

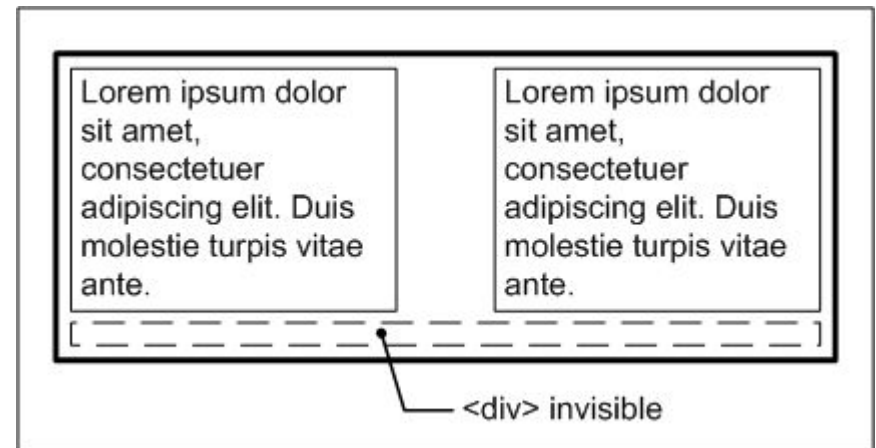
Los elementos que se encuentran alrededor de una caja flotante adaptan sus contenidos para que fluyan alrededor del elemento posicionado



# Posicionamiento flotante: problemática, limpiar floats



```
#contenedor {  
    border:    thick solid #000;  
    overflow: hidden;  
}
```



---

# Solapamiento de elementos (z-index)

- El atributo z-index permite definir el nivel de profundidad de una caja.
  - Su valor es un número entero.
  - El valor 0 suele tomarse como el nivel más bajo. Cuanto más alto sea el valor más cerca se mostrará la capa.
  - Z-index solo tiene efecto si aparece *position*
-

# CSS SPRITES

---

# CSS SPRITES

```
#home {  
  width: 46px;  
  height: 44px;  
  background: url(img_navsprites.gif) 0 0;  
}
```



[https://www.w3schools.com/css/tryit.asp?filename=trycss\\_sprites\\_nav](https://www.w3schools.com/css/tryit.asp?filename=trycss_sprites_nav)

# Formularios



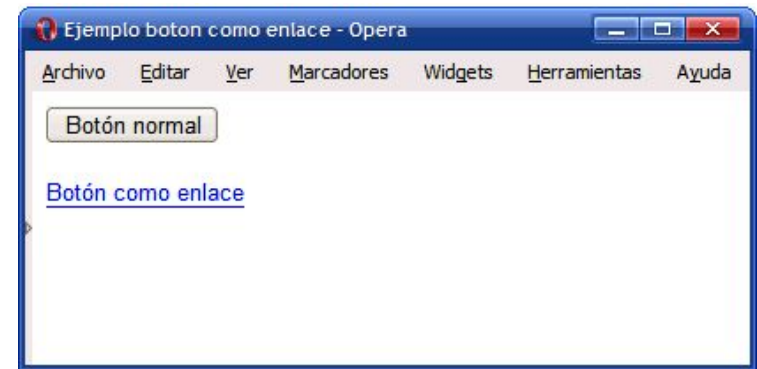
# Mostrar un botón como enlace

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

```
.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" />
```



# Mejoras en los campos de texto

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

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

Formulario sin padding en los input

Nombre

Contraseña

Formulario con padding en los input

Nombre

Contraseña

# Labels alineadas

- 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.

Alta en el servicio

Nombre  Apellidos

DNI

Contraseña

El diagrama muestra un formulario web con un título "Alta en el servicio". Los campos de entrada están alineados horizontalmente: "Nombre" y "Apellidos" están en la misma línea; "DNI" está a la derecha de un campo de entrada; "Contraseña" está a la derecha de un campo de entrada. Un botón "Dar de alta" está en la línea inferior.

# Labels alineadas

```
<form>
<fieldset>
  <legend>Alta </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">DNI</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>
```

# Labels alineadas

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

Alta en el servicio

Nombre

Apellidos

DNI

Contraseña


# Labels alineadas en la misma línea

- ```
<form>
  <fieldset>
    <legend>Alta en el servicio</legend>

    <div>
      <label for="nombre">Nombre</label>
      <input type="text" id="nombre" />

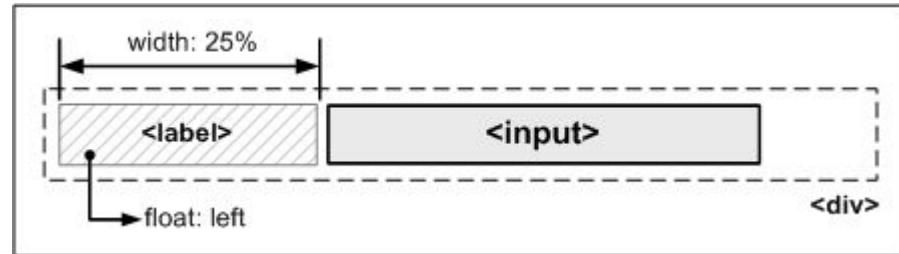
    </div>

    <div>
      <label for="apellidos">Apellidos</label>
      <input type="text" id="apellidos" size="35" />
    </div>

    ...
  </fieldset>
</form>
```
- Cada línea formada por label e input se incluye en una capa contenedor.
- 

# Labels alineadas en la misma línea

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



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 de alta"/>	

---

Precedencia de estilos.

---



# Precedencia de estilos

- La precedencia de estilos es una manera de indicar que un estilo definido prevalece por encima de otro definido en la misma o en CSS diferentes.
- Esto es necesario cuando hay dos o más estilos que actúan sobre los mismos atributos pero con diferente valor.
- La precedencia de estilos va asociado con el concepto de especificidad, que se refiere al peso de cada uno de los elementos de una hoja de estilo.

# Precedencia de estilos

- Un cálculo sencillo para calcular la especificidad de una regla es sumar los puntos según los selectores que contenga.
  - ❑ 1 punto: a un selector de etiqueta
  - ❑ 10 puntos: selector de clase
  - ❑ 100 puntos: selector de identificador.
  - ❑ 1.000 puntos: atributo de estilo (atributo style).
- Además de la especificidad, se puede utilizar la declaración *!important*. Se sitúa en el lado del valor del atributo, antes del ;.
  - ❑ Ej:

```
p{  
    background: #1199ff !important;  
    background: crimson;  
}
```

# Definición de hojas de estilo

---

# Hojas de estilo en cascada externa

```
<link rel=stylesheet href="fichero.css"  
title="estilo">
```

- El atributo *rel* se usa para definir la relación con el fichero enlazado:
  - *rel=stylesheet* especifica un estilo persistente o preferido.
    - Persistente es aquel que se aplica si están activas las hojas de estilo.
    - Preferido es uno que se aplica automáticamente. La combinación de *rel=stylesheet* y un atributo *title* especifica un atributo preferido. No se puede especificar más de un estilo preferido.
  - *rel="alternate stylesheet"* define un estilo alternativo que el usuario podría elegir para reemplazar la hoja de estilo preferido.

---

# Hojas de estilo en cascada externa

- Un estilo también puede definirse mediante múltiples hojas de estilo.

- Ej:

```
<link rel=stylesheet href="basico.css" title="estilo">
```

```
<link rel=stylesheet href="tablas.css" title="estilo">
```

```
<link rel=stylesheet href="formas.css" title="estilo">
```

---

---

# Hojas de estilo en cascada externa

- Otra alternativa es usar la regla `@import` incluida dentro de las etiquetas *style*.
  - Ejemplo  
`<style>@import url("estilos.css");</style>`
  - El funcionamiento es igual que el anterior aunque `@import` no está soportado por todos los navegadores.
-

# Herramientas y test de verificación

- W3C proporciona herramientas para validar código y hojas de estilo, comprobando si éstas son correctas según las gramáticas publicadas.
- <http://jigsaw.w3.org/css-validator/>
- Si la validación no encuentra errores, sus autores podrán incluir un icono como el siguiente indicando que los desarrolladores se han preocupado por crear un sitio web interoperable y acorde al estándar.

