



Tema 2. Tecnologías del cliente. CSS

Programación web

Boni García
Curso 2017/2018

Índice

1. HTML
2. CSS
3. Bootstrap
4. JavaScript
5. jQuery

Índice

1. HTML
2. CSS
 - Introducción
 - Formas de insertar CSS en HTML
 - Formato
 - Colores
 - Tamaño
 - Tipo de letra
 - Bordes
 - Ejemplo: botón básico con CSS
 - Selectores CSS
 - Reglas
 - Modelo de cajas
 - Más allá de CSS: Sass
3. Bootstrap
4. JavaScript
5. jQuery

Introducción

Cascading Style Sheets (CSS)

- Es un lenguaje utilizado para dar **estilo** a contenido estructurado
- Se aplica principalmente a documentos **HTML**, pero también se puede usar con otros documentos como
 - **SVG** (*Scalable Vector Graphics*): Gráficos vectoriales
 - **XML** (*eXtensible Markup Language*): Lenguaje de marcado
- Con **CSS** se pueden especificar:
 - **Colores**: principal, de fondo, degradados, etc.
 - **Tipografía**: familia, tamaños, etc.
 - **Layout**: Disposición de los elementos en el documento
 - **Efectos**: sombras, esquinas redondeadas, etc.

Introducción

Historia

- Inicialmente **HTML** contaba con muchas etiquetas destinadas a la **presentación** (estilo) y no a la **estructura**
- Para separar la presentación de la estructura se creó **CSS**. Ventajas:
 - Separación roles: diseñadores y programadores
 - Se pueden usar distintos CSS's en función del tipo del dispositivo
 - La página es más accesible porque el contenido está separado de la presentación
- La primera versión se estandariza en **1996** como **CSS 1**
- En el 2011 se publicó la versión **CSS 2.1**
- La versión **CSS 3** tiene algunas partes (módulos) finalizados y otros en desarrollo <http://www.w3.org/Style/CSS/current-work>

Formas de insertar CSS en HTML

1. Incluido en un elemento

- Mediante el atributo HTML `style`:

```
<!DOCTYPE html>
<html>
<body>
<p style="color: red">Red
text!</p>
</body>
</html>
```



Red text!

2. Incluido en la página

- En la sección `head` con la etiqueta `style`:

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
color: green;
}
</style>
</head>
<body>
<p>Green text!</p>
</body>
</html>
```



Green text!

3. En fichero independiente

- Enlazado con etiqueta `link` en sección `head`:

```
<!DOCTYPE html>
<html>
<head>
<link type="text/css"
rel="stylesheet"
href="styles.css">
</head>
<body>
<p>Blue text!</p>
</body>
</html>
```

styles.css

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



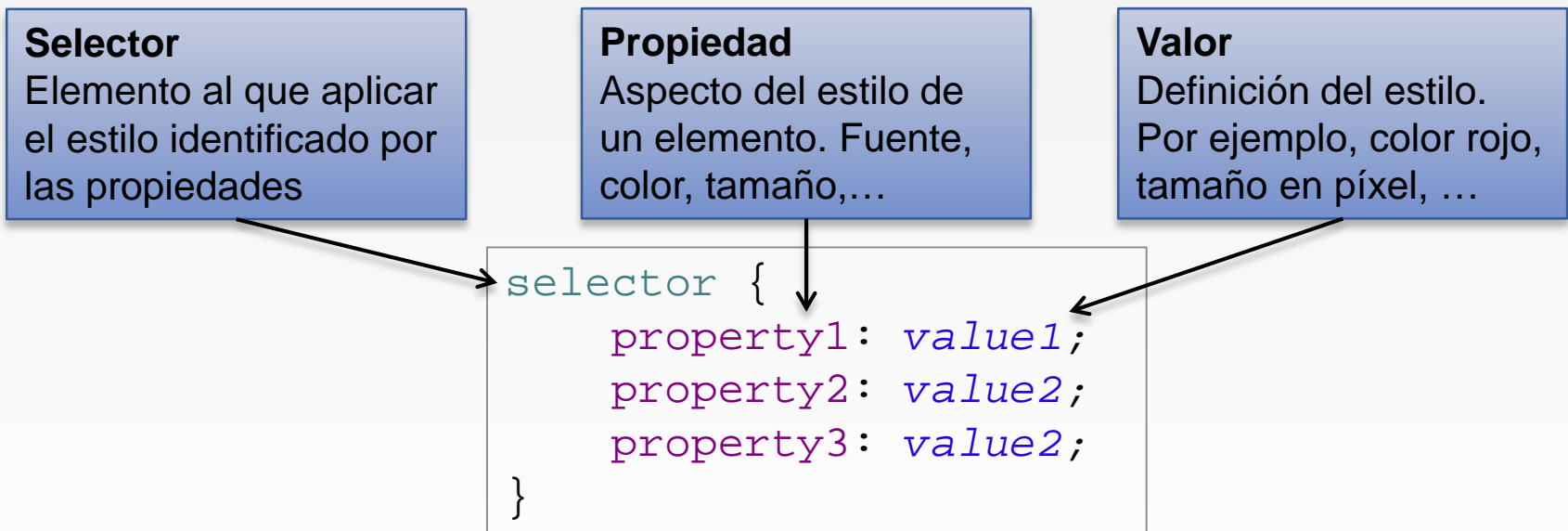
Blue text!

Formas de insertar CSS en HTML

- En general, los estilos CSS en un fichero independiente (opción 3) es la forma más recomendada
- Ventajas:
 - Un mismo fichero CSS se puede aplicar a **diferentes** páginas HTML
 - Ese fichero puede estar **cacheado** por el navegador para que no se descargue repetidamente
 - Se pueden incluir **varios** ficheros CSS asociados al mismo HTML

Formato

- Un documento CSS tiene el siguiente formato:



Formato

```
<h3>What's CSS for?</h3>
<p>CSS is for styling HTML
pages!</p>
<h3>Why use it?</h3>
<p>
It makes web pages look
<span>really cool</span>.
</p>
<h3>What do I think of it?</h3>
<p>It's awesome!</p>
```

```
p {
  font-family: Arial;
  color: blue;
  font-size: 12px;
}

h3 {
  color: red;
}

span {
  background-color: yellow;
}
```



What's CSS for?

CSS is for styling HTML pages!

Why use it?

It makes web pages look really cool.

What do I think of it?

It's awesome!

Formato

Cascading Style Sheets (CSS)

- Se dice que son en **cascada** porque a los elementos se les aplican algunas de las propiedades del elemento **padre** a no ser que se indique un estilo particular para dicho elemento
- Como un **estilo está compuesto** de varias **propiedades** (color, tamaño, tipo de letra...), es posible que se declaren algunas nuevas pero otras se sigan **heredando** del elemento **padre**
- En el ejemplo anterior, el elemento `` hereda el color azul y demás propiedades del elemento `<p>`

Formato

Efectos de texto

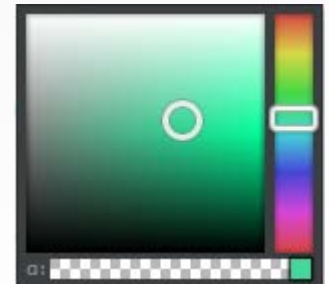
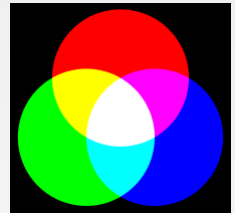
```
p {  
  text-align: center;  
  text-decoration: line-through underline overline;  
  text-transform: capitalize;  
}  
  
h1 {  
  text-shadow: 5px 5px 2px red;  
  text-indent: 50px;  
}
```



Fork me on GitHub

Colores

- Propiedad `color`: Color de texto
- Propiedad `background-color`: Color de fondo
- Valores de estas propiedades:
 - Nombre del color: *black, white, red, green, blue, yellow, purple...*
 - Color **RGB** en formato hexadecimal: *#000000 - #FFFFFF*
 - Color **RGB** en formato decimal: *rgb(0,0,0) - rgb(255,255,255)*
 - Color **RGB** con transparencia (*alpha*): *rgba(0,0,0,0.0) - rgba(255,255,255,1.0)*
- Más información:
 - http://devdocs.io/css/color_value



Tamaño

- Tamaño de texto: propiedad `font-size`
- Tamaño de elementos: propiedad `width`: (ancho) y `height` (alto)
- Formas de especificar el tamaño de los elementos HTML:
 - Unidad relativa: `em` (recomendada para texto, para propiciar la adaptación a diferentes dispositivos y resoluciones). Esta unidad es relativa al padre del elemento, mientras que `rem` es relativa al tamaño de inicial del documento (root)
 - `1em`: Tamaño de texto normal
 - `0.5em`: Tamaño de texto mitad de lo normal
 - `2em`: Tamaño de texto doble de lo normal
 - Unidad absolutas: `px`, `cm`, `mm`, `in`, `pt`, `pc`
 - Literales para fuentes: `medium`, `xx-small`, `x-small`, `large`, `x-large`, `xx-large`
- Más información: <http://devdocs.io/css/length>

Fondos

Fork me on GitHub

```
body {  
  background-image: url("gradient_bg.png");  
  background-repeat: repeat-x;  
}
```



Hello World

```
body {  
  background-image: url("css3.png");  
  background-repeat: no-repeat;  
  background-position: center;  
  background-attachment: fixed;  
}
```



inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eirmod consectetur signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.

Ius id vidit volumus mandamus, vide veritus democritum te nec, ei eos debet libris consulatu. No mei ferri graeco dicunt, ad cum veri accommodare. Sed at malis omnesque delicata, usu et iusto zzril meliore. Dicunt maiorum eloquentiam cum cu, sit summo dolor essent te. Ne quodsi nusquam legendos has, ea dicit voluptua eloquentiam pro, ad sit quas qualisque. Eos vocibus deserunt quaestio ei.

Blandit incorrupte quaerendum in quo, nibh impedit id vis, vel no nullam semper audiam. Ei populo graeci consulatu mei, has ea stet modus phaedrum. Inani

Tipo de letra

- Propiedad `font-family`: tipo de letra
- Depende de los tipos de letra instalados en el sistema operativo en el que se ejecuta el navegador
- Como no se puede saber qué tipos de letra están instalados lo que se suele hacer es especificar una **lista** de tipos de letra,
- Es recomendable que el último sea un tipo de letra **genérico** que siempre van a existir en un navegador: *serif*, *sans-serif*, *monospace*, *cursive*, *fantasy*
- Otras fuentes comunes: *Arial*, *Verdana*, *Times*
- Más información:
 - <http://devdocs.io/css/font-family>

Tipo de letra

- Tipos genéricos:

```
font-family: serif;  
font-family: sans-serif;  
font-family: monospace;  
font-family: cursive;  
font-family: fantasy;
```



This is an example of a serif font.

This is an example of a sans-serif font.

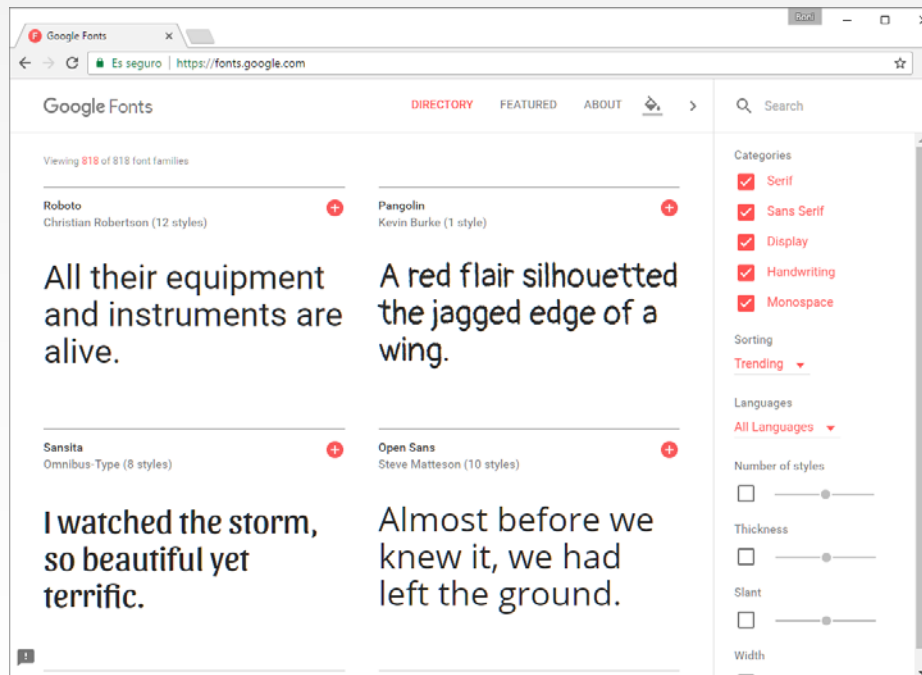
This is an example of a monospace font.

This is an example of a cursive font.

This is an example of a fantasy font.

Tipo de letra

- Es posible configurar una página web para descargar tipografías
- **Google Fonts** proporciona cientos de tipos de letra libres



<https://fonts.google.com/>

Bordes

- Hay muchos elementos en una página HTML que pueden tener bordes
- El borde se configura con la propiedad `border`
- Se pueden cambiar individualmente los bordes mediante las propiedades `border-top`, `border-left`, `border-right`, `border-bottom`
- Esta propiedad está formada por 3 valores:
 - **Ancho:** Típicamente en `px`
 - **Estilo:** `none`, `hidden`, `dotted`, `solid`, `dashed`...
 - **Color:** misma notación que hemos visto antes (RGB, literal)
- Para bordes redondeados se usa la propiedad `border-radius`: `Npx`;
- Más información:
 - <http://devdocs.io/css/border>

```
table {  
    border: 1px solid black;  
}
```

Ejemplo: botón básico con CSS

button.html

```
<!DOCTYPE html>
<html>

<head>
  <title>Facebook button</title>
  <link type="text/css" rel="stylesheet"
href="button.css">
</head>

<body>
  <div>
    <a href="#">Friend us on
<span>Facebook!</span></a>
  </div>
</body>

</html>
```

button.css

```
div {
  width: 120px;
  height: 50px;
  background-color: #BCD2EE;
  border: 5px solid #6495ED;
  border-radius: 15px;
  text-align: center;
}

a {
  text-decoration: none;
  color: #3D59AB;
  font-family: Verdana, sans-serif;
}

span {
  font-weight: bold;
  font-size: 18px;
  color: #ffffff;
}
```

Fork me on GitHub



Friend us on
Facebook!

Selectores CSS

- Los **selectores** sirven para seleccionar los elementos a los que se les aplicará el estilo en CSS
- Hasta ahora hemos visto que cuando se usa como selector el nombre de una **etiqueta**, esas propiedades se aplicarán a todos los elementos de ese tipo en la página

```
a {  
  color: #3D59AB;  
}
```

El estilo se aplicará a todos los **enlaces** de la página

- Existen otras formas de seleccionar elementos

Selectores CSS

Elementos dentro de elementos

- Elementos que están dentro de otros elementos:

```
<div>
  <div>
    <p>Hello world!</p>
  </div>
</div>
<p>Hi there!</p>
```

```
div>div>p {
  color: red;
}
```



Hello world!
Hi there!

El estilo se aplicará a todos los elementos `p` que estén dentro de un `div` que a su vez esté dentro de otro `div`

Selectores CSS

Elementos dentro de elementos

- Podemos seleccionar elementos dentro de otros elementos pero permitiendo que haya elementos entre medias
- Quitamos el símbolo >

```
<div>
  <ul>
    <li>One</li>
    <li>Two</li>
  </ul>
</div>
```

```
div li {
  /* CSS stuff! */
}
```

El estilo se aplicará a todos los elementos `li` que estén dentro de un `div` aunque no sea directamente

Selectores CSS

Selector comodín

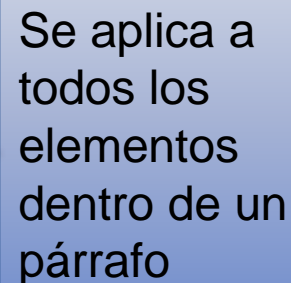
- Selector `*` para seleccionar cualquier elemento de la página:

```
* {  
  border: 2px solid black;  
}
```

- También se puede usar el selector comodín `*` para elementos dentro de otros elementos:

```
p * {  
  margin: 10px;  
}
```

Se aplica a todos los elementos dentro de un párrafo



Selectores CSS

- ¿Qué ocurre si a un mismo elemento le corresponden varias reglas CCS?
- En el siguiente ejemplo, a los párrafos le corresponde diferentes colores de texto:

```
ul li p {  
    color: red;  
}
```

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

- Siempre se aplica el **selector más específico**:

```
<p>Shopping cart:</p>  
<ul>  
  <li><p>Milk</p></li>  
  <li><p>Cookies</p></li>  
  <li><p>Sugar</p></li>  
</ul>
```

Shopping cart:

- Milk
- Cookies
- Sugar

Selectores CSS

Seleccionar por clase o por identificador

- Es habitual dar un nombre a un elemento concreto (atributo `id`)

```
<p id="main">My text</p>
```

```
#main {  
  color: #FF00FF;  
  font-weight: bold;  
}
```



My text

- También se puede crear un tipo de elementos (atributo `class`)

```
<p class="text">Other text</p>
```

```
.text {  
  color: red;  
}
```



Other text

Selectores CSS

Pseudo-clases

- Se pueden usar los selectores para aplicar estilos diferentes dependiendo del **estado** de un elemento (**pseudo-clases**)

```
selector:pseudo-class_selector {  
    property: value;  
}
```

- Por ejemplo, las pseudo-clases de un enlace son (el orden importa):
 - `a:link`: Estilo del enlace no visitado
 - `a:visited`: Estilo del enlace visitado
 - `a:hover`: Estilo del enlace cuando se pasas el ratón por encima
 - `a:active`: Estilo del enlace activo (al hacer click)

Reglas

- La regla `@import` sirve para importar hojas de estilos externas
- La regla `@font-face` sirve para definir fuentes en base a una tipografía externa
- La regla `!important` sirve para asegurar que un determinado estilo se aplique siempre

```
@import "button.css";  
@font-face {  
  font-family: "sansation";  
  src: url(sansation_light.woff);  
}  
p {  
  font-family: sansation;  
  color: #ff0000 !important;  
}  
p {  
  color: #000000;  
}
```



This is a link
This is a paragraph

Reglas

- La regla `@media` sirve para definir un conjunto de estilos CSS que sólo se aplica cuando se cumple una determinada condición (**media query**)
- Entre otras funciones, se usan para implementar diseño adaptable (*responsive design*)

Tamaño de pantalla

Estilos de impresión

```
@media screen and (min-width: 600px) {
  /* One or more CSS styles to apply when
  the query is satisfied */
}
```

```
@media screen and (max-width: 1200px) {
  /* One or more CSS styles to apply when
  the query is satisfied */
}
```

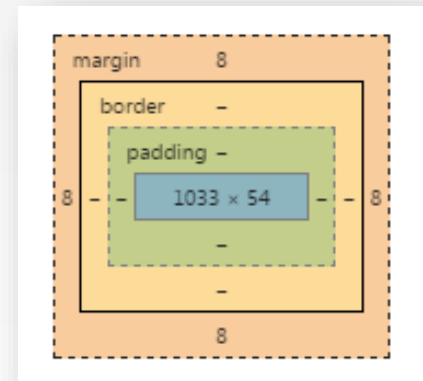
```
@media print {
  /* CSS styles to apply when the query
  is satisfied */
}
```

```
@media (orientation : portrait) {
  /* CSS styles to apply when the query
  is satisfied */
}
```

Orientación de la pantalla

Modelo de cajas

- El modelo de cajas (***box model***) hace que todos los elementos de las páginas HTML se representen mediante cajas rectangulares
- Cada elemento está en una caja que tiene:
 - **Margen:** Espacio transparente alrededor del elemento
 - **Borde:** Línea que rodea el elemento
 - **Relleno (*padding*):** Relleno del elemento desde su contenido hasta el borde
 - **Contenido:** Información del propio elemento
 - **Borde exterior:** Línea tras el margen



Modelo de cajas

Propiedad `margin`

- Controla el espacio alrededor de un elemento
- `margin: auto` : Pone el mismo margen a la izquierda y a la derecha. Es decir, centra el elemento horizontalmente
- `margin-top`, `margin-right`, `margin-bottom`, `margin-left`: Permiten cambiar el margen individualmente a cada lado
- `margin: 1px 2px 3px 4px` : Forma compacta de cambiar todos los márgenes a la vez (top right bottom left)
- `margin: 1px 2px` : Otra forma compacta de cambiar todos los márgenes a la vez (top-bottom left-right). Ejemplo típico:
 - `margin: 10px auto` : Deja 10px arriba y abajo y centra el contenido
- `margin: 1px` : El mismo margen a cada lado

Modelo de cajas

Propiedad `border`

- Controla el borde del elemento
- `border-width`: Ancho del borde
- `border-color`: Color del borde
- `border-style`: Estilo del borde: *solid*, *dashed*, *dotted*, ...
- `border`: *1px solid black* : Forma compacta de cambiar todas las propiedades a la vez

Modelo de cajas

Propiedad `padding`

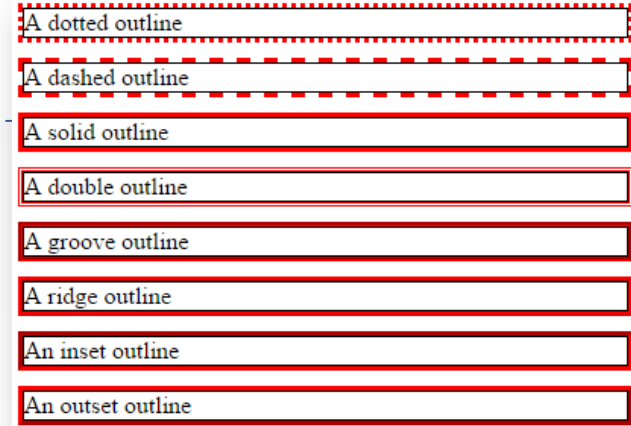
- Rellena entre el contenido y el borde
- `padding-top`, `padding-right`, `padding-bottom`, `padding-left`:
Permiten cambiar el relleno individualmente a cada lado
- `padding`: `1px 2px 3px 4px` : Forma compacta de cambiar todos los rellenos a la vez (top right bottom left)
- `padding`: `1px 2px` : Segunda forma compacta de cambiar todos los rellenos a la vez (top-bottom left-right)
- `padding`: `1px` : Forma compacta de poner el mismo relleno en todos los lados

Modelo de cajas

Propiedad `outline`

- Esta propiedad establece el tipo del borde exterior del modelo de cajas

```
p {border: 1px solid black; outline-color: red;}  
p.dotted {outline-style: dotted;}  
p.dashed {outline-style: dashed;}  
p.solid {outline-style: solid;}  
p.double {outline-style: double;}  
p.groove {outline-style: groove;}  
p.ridge {outline-style: ridge;}  
p.inset {outline-style: inset;}  
p.outset {outline-style: outset;}
```



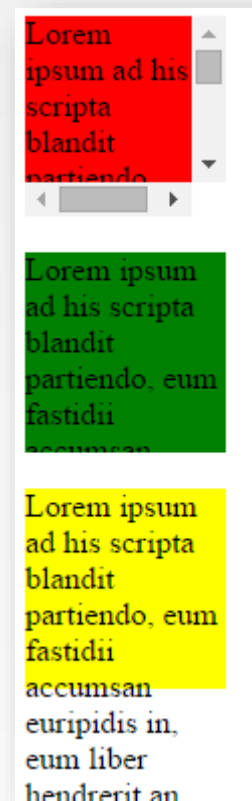
Modelo de cajas

Propiedad overflow

- Establece el comportamiento del contenido que se sale de una caja:

```
<div class="scroll">Lorem  
ipsum...</div>  
<br>  
<div class="hidden">Lorem  
ipsum...</div>  
<br>  
<div class="visible">Lorem  
ipsum...</div>
```

```
.scroll {  
  background-color: red;  
  width: 100px;  
  height: 100px;  
  overflow: scroll;  
}  
.hidden {  
  background-color: green;  
  width: 100px;  
  height: 100px;  
  overflow: hidden;  
}  
.visible {  
  background-color: yellow;  
  width: 100px;  
  height: 100px;  
  overflow: visible;  
}
```

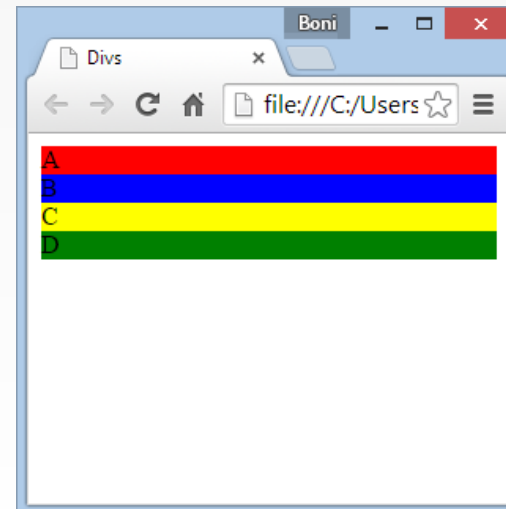


Modelo de cajas

- El modelo de capas aplica a **todos** los elementos HTML
- Vamos a estudiarlo aplicado a elementos `<div>`, dada su importancia para la composición (*layout*) de páginas web
- Por defecto, los elementos `<div>` ocupan todo el ancho de la página

```
#one { background-color: red }  
#two { background-color: blue }  
#three { background-color: yellow }  
#four { background-color: green }
```

```
<div id="one">A</div>  
<div id="two">B</div>  
<div id="three">C</div>  
<div id="four">D</div>
```



Modelo de cajas

Propiedad `display`

- Esta propiedad establece el tipo de caja de un elemento. Valores más importantes:
- `display: block` (opción por defecto)
 - No permite otros elementos en la misma línea
 - Se puede cambiar su alto y ancho
- `display: inline-block`
 - Permite otros elementos en la misma línea
 - Se puede cambiar su alto y ancho
- `display: inline`
 - Permite otros elementos en la misma línea
 - No se puede cambiar su alto y ancho
- `display: none`
 - Oculto

```
div {
  width: 50px;
  display: block;
}
```



```
div {
  width: 30px;
  display: inline-block;
}
```



```
div {
  width: 50px;
  display: inline;
}
```



```
div {
  width: 50px;
  display: none;
}
```

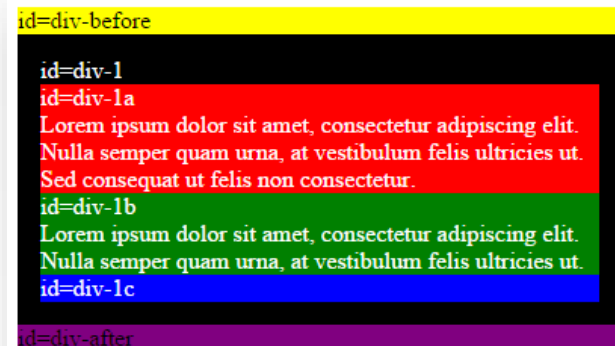
Modelo de cajas

- Vamos a usar el siguiente ejemplo para estudiar el posicionamiento:

Fork me on GitHub

```
#div-before {
  background-color: yellow;
  width: 400px;
}
#div-after {
  background-color: purple;
  width: 400px;
}
#div-1 {
  color: white;
  background-color: black;
  width: 370px;
  padding: 15px;
}
#div-1a {
  background-color: red;
}
#div-1b {
  background-color: green;
}
#div-1c {
  background-color: blue;
}
```

```
<div id="div-before">id=div-before</div>
<div id="div-1">
  id=div-1
  <div id="div-1a">
    id=div-1a <br>Lorem ipsum ...
  </div>
  <div id="div-1b">
    id=div-1b <br>Lorem ipsum ...
  </div>
  <div id="div-1c">id=div-1c</div>
</div>
<div id="div-after">id=div-after</div>
```

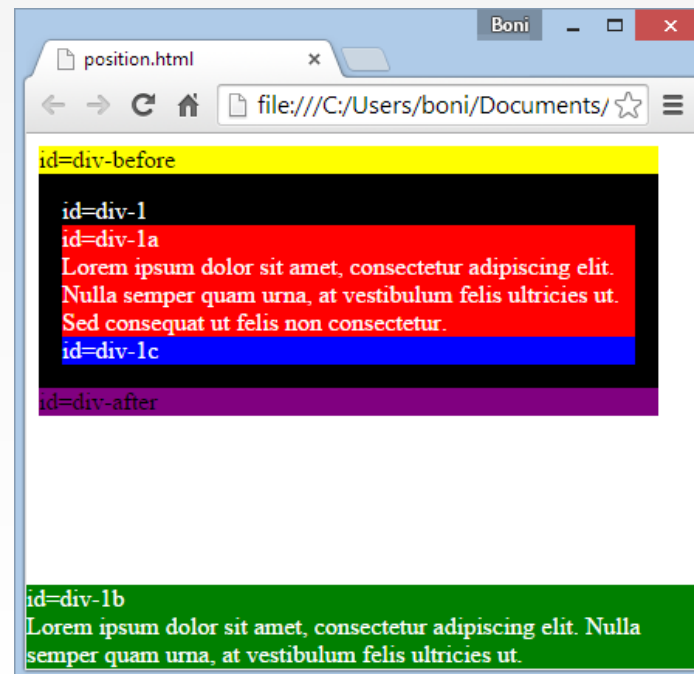


Modelo de cajas

Propiedad `position`

- Con la propiedad `position` podemos controlar la posición de un elemento dentro de la página
- `position: fixed;`
 - Fija un elemento a la pantalla
 - No se mueve aunque se haga scroll
 - Su posición se especifica con `top`, `bottom`, `left` o `right`

```
#div-1b {
  position: fixed;
  bottom: 0px;
  left: 0px;
}
```

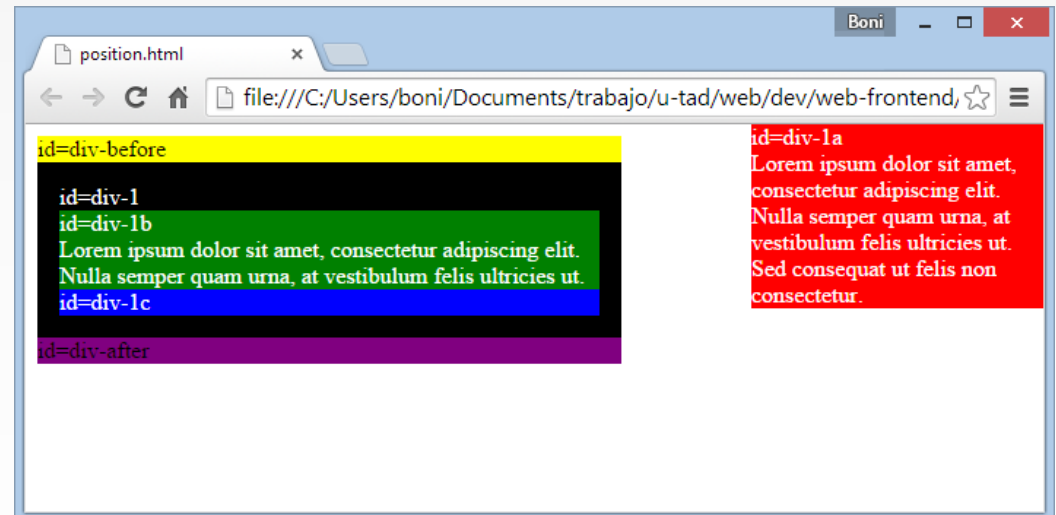


Modelo de cajas

Propiedad `position`

- `position: absolute;`
 - Se posiciona de forma absoluta en la posición indicada por las propiedades `top`, `bottom`, `left` o `right`
 - El elemento se extrae de su posición inicial

```
#div-1a {
  position: absolute;
  top: 0;
  right: 0;
  width: 200px;
}
```



Modelo de cajas

Propiedad `position`

- `position: relative;`
 - Mediante este posicionamiento movemos un elemento respecto a su posición por defecto (`static`)
 - Se usan las propiedades `top`, `bottom`, `left` o `right` para mover el elemento de forma relativa
 - A diferencia del posicionamiento absoluto, se mantiene la posición donde iba originalmente

```
#div-1 {
  position: relative;
  top: 20px;
  left: -40px;
}
```

id=div-before

```
=div-1
=div-1a
rem ipsum dolor sit amet, consectetur adipiscing elit.
lla semper quam urna, at vestibulum felis ultricies ut.
d consequat ut felis non consectetur.
=div-1b
rem ipsum dolor sit amet, consectetur adipiscing elit.
lla semper quam urna, at vestibulum felis ultricies ut.
=div-1c
```

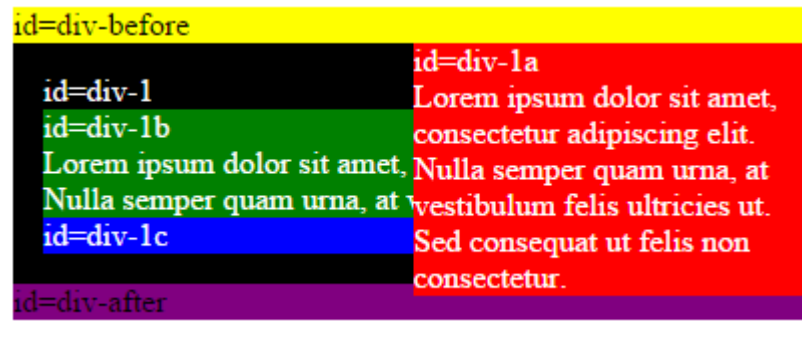

Modelo de cajas

Propiedad `position`

- Si un elemento tiene `position: relative`, los elementos que tenga dentro con `position: absolute` le tomarán como referencia en el posicionamiento absoluto

```
#div-1 {
  position: relative;
}

#div-1a {
  position: absolute;
  top: 0;
  right: 0;
  width: 200px;
}
```

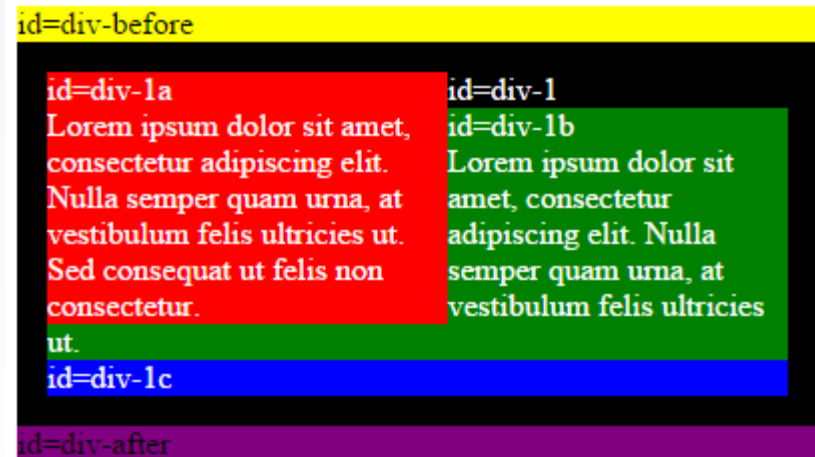


Modelo de cajas

Propiedad `float`

- Para capas de altura variable (lo normal en casi todas las páginas web) el posicionamiento absoluto no es una buena solución
- En su lugar se usa posicionamiento **flotante**
- Si un elemento tiene la propiedad `float: left` o `float: right` se desplaza a la izquierda o derecha respectivamente de la posición que originalmente ocupaba

```
#div-1a {
  float: left;
  width: 200px;
}
```

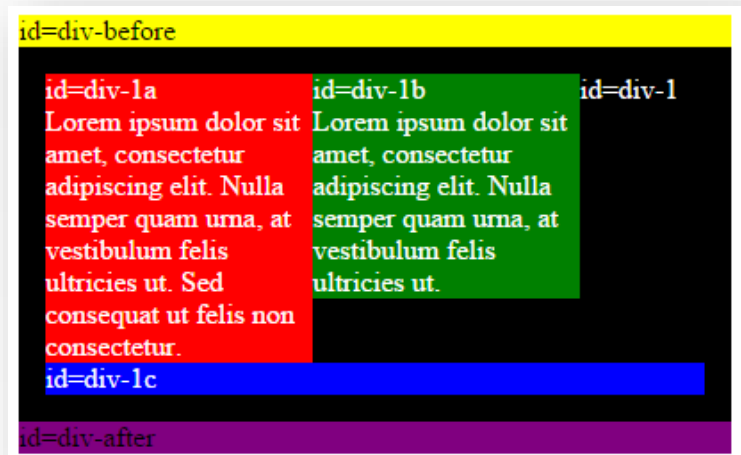


Modelo de cajas

Propiedad `clear`

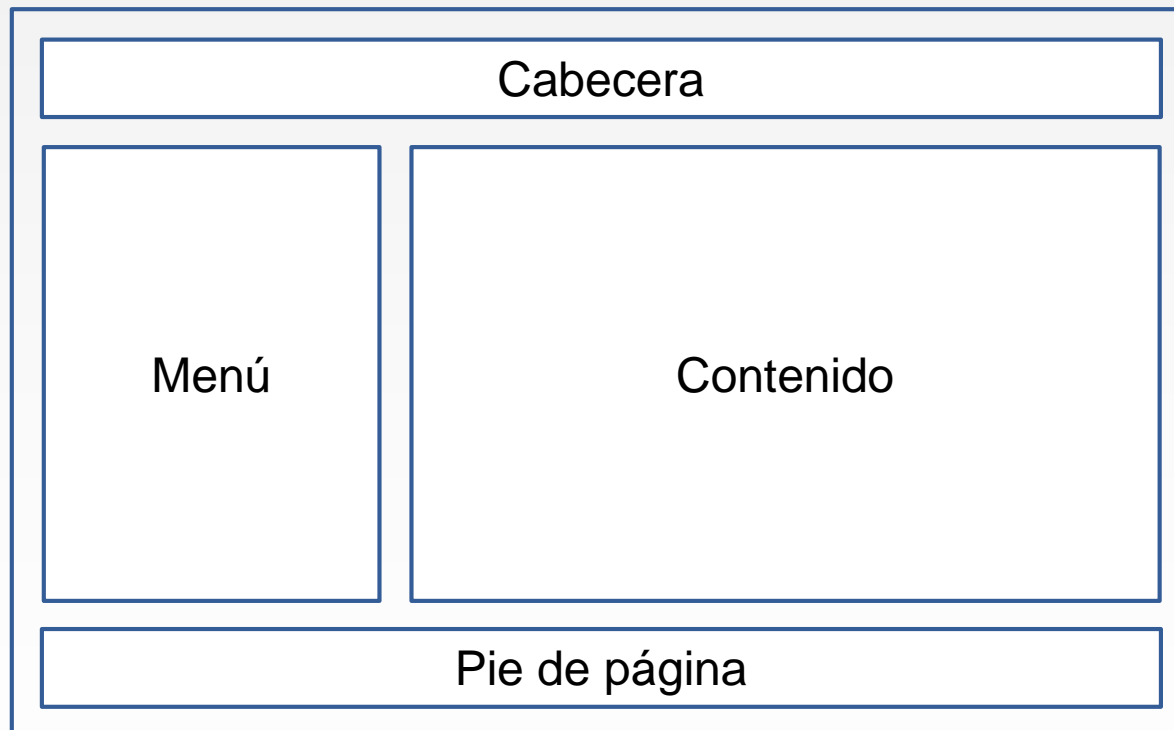
- La propiedad `clear` permite resetear el posicionamiento flotante, de forma que los siguientes elementos se muestren debajo de elementos flotantes
- Posibilidades:
 - `clear: left` : Elemento se sitúa debajo del anterior `float: left`
 - `clear: right` : Elemento se sitúa debajo del anterior `float: right`
 - `clear: both` : Elemento se sitúa debajo del que tenga mayor

```
#div-1a {
  float: left;
  width: 150px;
}
#div-1b {
  float: left;
  width: 150px;
}
#div-1c {
  clear: both;
}
```



Modelo de cajas

Ejemplo de *layout*

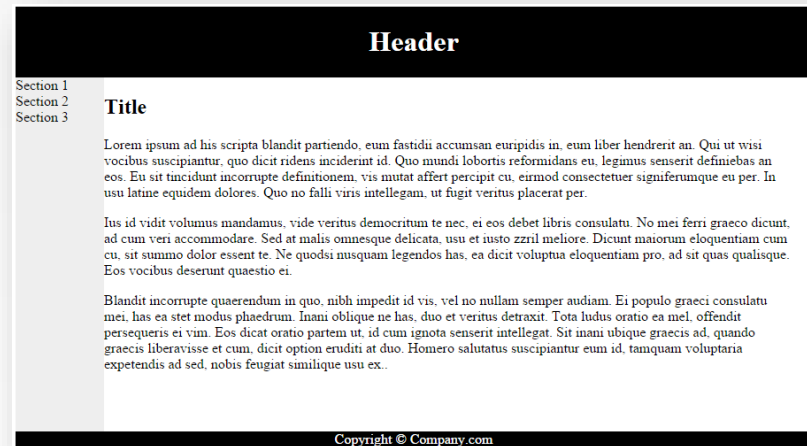


Modelo de cajas

Ejemplo de *layout*

```
.container { width: 900px; margin: 0 auto; }
.header { float: left; width: 100%; background-color: black; color: white; text-align: center; }
.nav { float: left; width: 100px; height: 400px; background-color: #eeeeee; }
.main { width: 780px; float: left; }
.footer { clear: both; background-color: black; color: white; text-align: center; }
```

```
<div class="container">
  <div class="header">
    <h1>Header</h1>
  </div>
  <div class="nav">
    Section 1<br> Section 2<br> Section 3
  </div>
  <div class="main">
    <h2>Title</h2>
    <p>Lorem ipsum ...</p>
  </div>
  <div class="footer">Copyright &copy;
    Company.com</div>
</div>
```



Fork me on GitHub

Modelo de cajas

Ejemplo de *layout*

```
.container { width: 900px; margin: 0 auto; }
.header { float: left; width: 100%; background-color: black; color: white; text-align: center; }
.content { float: left; width: 100%; position: relative; }
.nav { width: 100px; position: absolute; top: 0; left: 0; bottom: 0; background-color: #eeeeee; }
.main { margin-left: 100px; float: left; }
.footer { clear: both; background-color: black; color: white; text-align: center; }
```

```
<div class="container">
  <div class="header">
    <h1>Header</h1>
  </div>
  <div class="content">
    <div class="nav">
      Section 1<br> Section 2<br> Section 3
    </div>
    <div class="main">
      <h2>Title</h2>
      <p>Lorem ipsum ...</p>
    </div>
  </div>
  <div class="footer">Copyright &copy;
    Company.com</div>
</div>
```

Header

Section 1
Section 2
Section 3

Title

Lorem ipsum ad his scripta blandit partiendo, eum fastidii accusan euripidis in, eum liber hendrerit an. Qui ut wisii vocibus suscipiantur, quo dicit ridens inciderint id. Quo mundi lobortis reformidans eu, legimus senserit definiebas an eos. Eu sit tincidunt incorrupte definitionem, vis mutat affert percipit cu, eimod consetctetur signiferumque eu per. In usu latine equidem dolores. Quo no falli viris intellegam, ut fugit veritus placerat per.

Ius id vidit volumus mandamus, vide veritus democritum te nec, ei eos debet libris consulatu. No mei ferri graeco dicitur, ad cum veri accommodare. Sed at malis omnesque delicata, usu et iusto zrril meliore. Dicitur maiorum eloquentiam cum cu, sit summo dolor essent te. Ne quodsi nusquam legendos has, ea dicit voluptua eloquentiam pro, ad sit quas qualisque. Eos vocibus deserunt quaestio ei.

Blandit incorrupte quaerendum in quo, nibh impedit id vis, vel no ullam semper audiam. Ei populo graeci consulatu mei, has ea stet modus phaedrum. Inani oblique ne has, duo et veritus detraxit. Tota ludus oratio ea mel, offendit persequeris ei vim. Eos dicit oratio partem ut, id cum ignota senserit intellegat. Sit mani ubique graecis ad, quando graecis liberavisse et cum, dicit option eruditi at duo. Homero salutatus suscipiantur eum id, tamquam voluptaria expetendis ad sed, nobis feugiat similique usu ex.

Copyright © Company.com

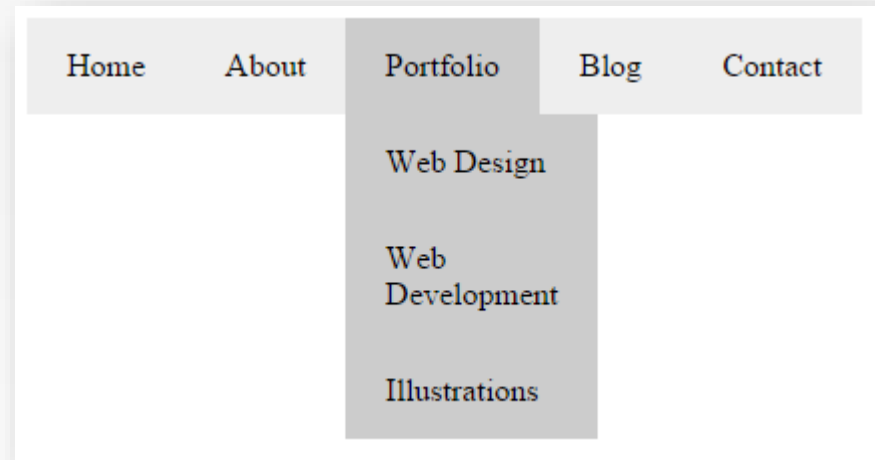
Fork me on GitHub

Modelo de cajas

Ejemplo de menú desplegable (*dropdown*)

```
ul li { display: inline-block; position: relative; margin-right: -4px; padding: 15px 20px; background: #eee; }
ul li:hover { background: #ccc; }
ul li ul { position: absolute; left: 0; top: 48px; padding: 0; display: none; }
ul li:hover ul { display: block; }
ul li ul li { background: #ccc; display: block; }
ul li ul li:hover { background: #eee; }
```

```
<ul>
  <li>Home</li>
  <li>About</li>
  <li>Portfolio
    <ul>
      <li>Web Design</li>
      <li>Web Development</li>
      <li>Illustrations</li>
    </ul>
  </li>
  <li>Blog</li>
  <li>Contact</li>
</ul>
```



Más allá de CSS: Sass

- **Sass** (*Syntactically Awesome Stylesheets*), que es un metalenguaje de CSS que enriquece la forma de dar estilos
- Se dice que Sass es un azúcar sintáctico que permite:
 - Definir variables
 - Anidamiento
 - Herencia de los selectores
- Sass se procesa a CSS
- Está escrito en Ruby (se instala como una gema)

```
gem install sass
```

<http://sass-lang.com/>



Más allá de CSS: Sass

- Sass propone dos sintaxis diferentes:
 - Sintaxis indentada. Extensión .sass
 - SCSS (Sassy CSS). Extensión .scss

