



# CSS3

© 2017, ACTIBYTI PROJECT SLU, Barcelona  
Autor: Ricardo Ahumada



MINISTERIO  
DE ENERGÍA, TURISMO  
Y AGENDA DIGITAL

**red.es**



ESTRATEGIA DE  
EMPRENDIMIENTO Y  
EMPLEO JUVENIL  
*garantía juvenil*



**UNIÓN EUROPEA**

Fondo Social Europeo  
*"El FSE invierte en tu futuro"*

1

# Introducción

# ¿Qué es CSS?

- Reciben el nombre de hojas de estilo en cascada o CSS (siglas en inglés de cascading style sheets) las indicaciones de estilo escritas es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML5.
- **Sintaxis**
  - El "destino" (a quién se aplica una regla) se establece mediante un **selector**.
  - Una indicación de estilo recibe el nombre de **regla de estilo**. Todas las reglas van entre {llaves} y terminan en ";".

**Selector:** indica que el estilo se aplicará a todos los tags <p>



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

**Regla de estilo:** indica el color de texto rojo.

- El World Wide Web Consortium (W3C) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores
  - <http://www.w3.org/Style/CSS/current-work>

# Ventajas de usar CSS 3

## ➤ Flexibilidad

- Principio de la separación de responsabilidades (separation of concerns)
- Los cambios en paletas de diseño, tipos de letras, tipo de formato, la imagen corporativa, etc.

## ➤ Programación

- Soporte del DOM y algunos de los nuevos atributos
- API que permite la creación, modificación, reasignación y manipulación dinámicas de los elementos CSS vinculados a cualquier elemento.

## ➤ Accesibilidad

- Posibilidad de acceder y modificar el paquete de nuevos atributos del conjunto WAI-ARIA desde CSS
- Permite crear páginas con distintos grados de accesibilidad
- Selección dinámica de la hoja de estilo que más se ajuste al nivel requerido por el lector.

# Ventajas de usar CSS 3

## ➤ Personalización

- Combinar las posibilidades de la gestión de cookies y establecer modificaciones iniciales
- Aspecto individualizado en función de preferencias de usuario.

## ➤ Consistencia entre sitios

- Una hoja de estilo podemos reutilizarla.
- Diseños auto-contenidos, en pequeñas unidades bien probadas.

## ➤ Economía del ancho de banda

- Reducir muchísimo la verbosidad para expresar los mismos requisitos con etiquetas HTML.

## ➤ Lenguaje declarativo y referencial

- Totalmente soportado por Visual Studio 2010

# Aplicando CSS

- Inline

```
<p style="color: red">text</p>
```

- En la página

```
<!DOCTYPE html>
<html>
<head>
<title>CSS Example</title>
<style>

    p {color: red;}

    a {color: blue;}

</style>
...
```

- Se recomienda no usar estas opciones, ya que acoplan el HTML con el estilo gráfico

# Aplicar CSS Externo (preferido)

- Se lleva el css a un archivo externo.
  - Este se referencia desde el HTML

```
<!DOCTYPE html>
<html>
<head>
    <title>CSS Example</title>
    <link rel="stylesheet" href="style.css">
...

```

- Fichero style.css

```
p {
    color: red;
}

a {
    color: blue;
}
```

# Selectores, propiedades y valores

- Los selectores son nombres asignados a los estilos en las hojas de estilo. De momento etiquetas HTML,
  - por ejemplo *p*, *div*, *header*....
- Para cada selector hay "propiedades" entre llaves, que toman la forma de palabras como
  - *color*, *font-weight* or *background-color*.
- Se da un valor a cada propiedad después de dos puntos (NO un signo "igual"). Los punto y coma ";" se utilizan para separar las propiedades.

```
body {  
    font-size: 14px;  
    color: navy;  
}
```

# Tamaños y porcentajes

Hay unidades generales para los valores usados en CSS para expresar dimensiones como tamaño, ancho, alto.

- **px** (como en font-size: 12px) es la unidad de píxeles.
- **em** (como en font-size: 2em) es la unidad para el tamaño calculado en función del tamaño base de fuente. Así por ejemplo "2em", indica dos veces el tamaño de fuente actual.
- **pt** (como font-size: 12pt) es la unidad para puntos, para medidas típicamente en medios impresos.
- **%** (Como en width: 80%) es la unidad de porcentajes relativos al ancho o alto de la página o del elemento padre.

# Colores

- CSS tiene 16.777.216 colores a tu disposición.
- Pueden tomar la forma de un nombre, un valor RGB (rojo / verde / azul) o un código hexadecimal.
- Por ejemplo, para indicar rojo, se puede usar:
  - red
  - rgb(255,0,0)
  - rgb(100%,0%,0%)
  - rgba(255,0,0,0)
  - #ff0000
  - #f00
- Los colores se pueden aplicar usando color y background-color.

```
h1 {  
    color: #ffc;  
    background-color: #009;  
}
```

# Texto

Puedes cambiar el tamaño y la forma del texto en una página web con un rango de propiedades.

- **font-family:** familia de fuentes como Times New Roman, Arial, or Verdana. → font-family: "Times New Roman"
- **font-size:** Tamaño de fuente
- **font-weight:** El grado de negrita de una fuente → font-weight: bold
  - Valores: bold, normal, bolder, lighter, 100, 200, 300, 400 (same as normal), 500, 600, 700 (same as bold), 800 or 900.
- **font-style:** itálica o no → font-style: italic; font-style: normal.
- **text-decoration:** subrayado debajo, encima o atravesado → text-decoration: underline
- **text-transform:** Cambia el tamaño (mayúsculas o minúsculas) del texto → text-transform: capitalize

# Texto Ejemplo

- Todo junto se puede ver así:

**font-weight: bold**

*font-style: italic*

FONT-VARIANT: SMALL-CAPS

TEXT-TRANSFORM: UPPERCASE

```
body {  
    font-family: arial, helvetica, sans-serif;  
    font-size: 14px;  
}  
  
h1 {font-size: 2em;}  
  
h2 {font-size: 1.5em;}  
  
a {text-decoration: none;}  
  
strong {  
    font-style: italic;  
    text-transform: uppercase;  
}
```

# Espaciado de texto

- letter-spacing: espaciado entre letras
- word-spacing : espaciado entre palabras
- line-height: Alto de línea
- text-align: Alineado a izquierda, derecha, centro
- text-indent: Indentado de la primera línea de un párrafo

Let's get some  
indenting and word-spacing  
going on.

And how about some  
letter-spacing, line-  
height, and justified  
text-alignment?

```
p {  
    letter-spacing: 0.5em;  
    word-spacing: 2em;  
    line-height: 1.5;  
    text-align: center;  
}
```

# El modelo de caja

- El modelo de caja funciona así: en el medio tienes el área de contenido (digamos una imagen), rodeándolo tienes el padding, rodeándolo tienes el borde y el entorno tienes el margen.
- Se puede representar visualmente así:



- No necesitas usar todos los atributos, pero este modelo funciona para todos los elementos de la página.

# Margen y padding

- Margen y padding son las propiedades más comúnmente usadas para espaciar los elementos de una página.
- Un margen es el espacio fuera de un elemento, mientras que el padding es el espacio dentro de algo.

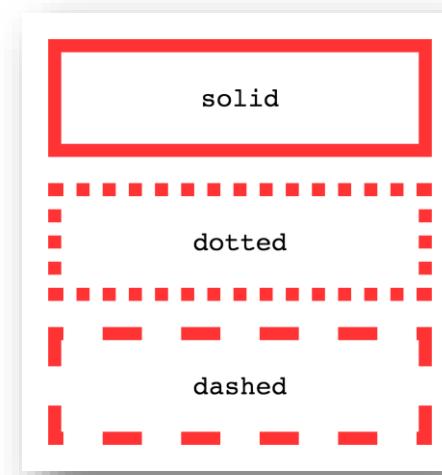
```
h2 {  
    font-size: 1.5em;  
    background-color: #ccc;  
    margin: 20px;  
    padding: 40px;  
}
```

# Bordes

Genera un borde alrededor de un elemento

- border-style: Estilo de borde
  - solid, dotted, dashed, double, groove, ridge, inset and outset
- border-width: ancho de borde. Existen propiedades específicas para cada lado:
  - border-top-width, border-right-width, border-bottom-width and border-left-width
- border-color: el color de borde
- La versión resumida sería
  - border: ancho estilo color → border: 2px dashed #ccc;

```
h2 {  
    border-style: dashed;  
    border-width: 3px;  
    border-left-width: 10px;  
    border-right-width: 10px;  
    border-color: red;  
}
```





# Pongámoslo en práctica

- Genera el HTML y CSS para la siguiente página
- <http://www.webstepbook.com/supplements/labsection/lab1-aboutme/images/aboutme.png>

**About Victoria Kirst**

*My name is Victoria and I am jolly, clumsy, and four-eyed.*

**My Classes This Quarter**

- CSE 451 - Operating Systems
- CSE 471 - Computer Design and Organization
- PHYS 121 - Physics: Mechanics
- CSE 498 - Research w/ Prof. Luis Ceze

**My Favorite Movies**

*(I actually don't watch too many movies, so...here goes!)*

1. The last 30 minutes of Forrest Gump ([MDB](#))
2. Star Trek Episode V with Zazu ([MDB](#))
3. Fight Club (not really, but I've seen like 3 movies total so this is my 3rd fave by technicality) ([MDB](#))

**My Moods**

**Fun Facts About My Neighbors**

- Sue Smith: *Effervescent* is a word that describes her.
- Bill Thompson: Loves playing *Yu-Gi-Oh*.

# Selectores de clase y por identificador

- Se pueden definir selectores propios en forma de selectores de clase e ID.
- Tiene el beneficio de poder presentar el mismo elemento HTML de manera diferente dependiendo de su clase o ID.
- Usaremos un **ID** para identificar un elemento concreto, y una **clase** se para identificar elementos con las mismas características.

```
<div id="top">  
  
  <h1>Chocolate curry</h1>  
  
  <p class="intro">This is my recipe for making  
  curry purely with chocolate</p>  
  
  <p class="intro">Mmm mm mmmmm</p>  
  
</div>
```

```
#top {  
  background-color: #ccc;  
  padding: 20px  
}  
  
.intro {  
  color: red;  
  font-weight: bold;  
}
```

# Selectores de clase y por identificador

- Se puede definir selectores que apliquen a clases de un tipo de elemento concreto
- Se indican mediante el punto “.”
  - tag.`una_clase` {reglas} → selecciona todas las etiquetas con atributo class = una\_clase
  - Por ejemplo:

```
p.intro {...} /*todos los <p> de clase intro*/  
div.subrayado {...} /*todos los <div> de clase subrayado*/
```

- Puede ser general
  - `.una_clase` {reglas} → selecciona cualquier elemento con atributo class = una\_clase
  - Por ejemplo:

```
p.intro {...} /*todos los <p>,<div>,<section>... de clase intro*/
```

# Agrupamiento

- Puede dar las mismas propiedades a una serie de selectores sin tener que repetirlos, mediante una coma “,”

```
h2, .thisOtherClass, .yetAnotherClass {  
    color: red;  
}
```

# Anidamiento

- Si el CSS está bien estructurado, no debería ser necesario usar muchos selectores de clase o ID. Esto se debe a que puede especificar propiedades para selectores dentro de otros selectores.

```
#top {  
    background-color: #ccc;  
    padding: 1em  
}  
  
#top h1 {  
    color: #ff0;  
}  
  
#top p {  
    color: red;  
    font-weight: bold;  
}
```

# Imágenes de fondo

- › Son una manera potente de agregar una presentación detallada a una página.

```
body {  
    background: white url(http://www.htmldog.com/images/bg.gif) no-repeat top right;  
}  
  
div{  
    url("bg.jpg") 20% 0% / contain repeat-y fixed content-box padding-box #fff  
}
```

- › El código anterior fusiona estas propiedades:
  - › background-color
  - › background-image, ubicación de la propia imagen.
  - › background-repeat, cómo se repite la imagen:
    - › repeat, efecto “tile” en todo el background,
    - › repeat-y, repetir verticalmente,
    - › repeat-x repetir horizontalmente
    - › no-repeat no se repite.
  - › background-position, que puede ser top, center, bottom, left, right, un tamaño, un porcentaje, o combinaciones como top right.
  - › background-size, el tamaño de la imagen: auto, tamaño, %, contain, cover, ancho alto

# Display

Indica cómo se mostrará un bloque

- Inline: caja en linea → li { display: inline }



- block: caja independiente → #navigation a {display: block;}
- inline-block: Mantendrá un cuadro en línea pero prestará mayor flexibilidad de formato a los bloques; por ejemplo, permitiendo margen a la derecha e izquierda de la caja.
- none: no muestra el bloque



2

## Layout de página

# Posicionamiento (position)

La propiedad de posición se utiliza para definir si un bloque es absoluto, relativo, estático o fijo:

- **static** es el valor predeterminado y representa un cuadro en el orden normal de las cosas, tal y como aparecen en el HTML.
- **relative** es muy similar a la estática, pero la caja puede ser desplazada desde su posición original con las propiedades superior, derecha, inferior e izquierda.
- **absolute** saca una caja del flujo normal del HTML y lo posiciona de manera absoluta. El bloque se puede colocar en cualquier parte de la **página** usando la parte superior, derecha, inferior e izquierda.
- **fixed** se comporta como absolute, pero posicionará absolutamente una caja en referencia a la ventana del navegador en lugar de la página web, por lo que los bloques fijos deben permanecer exactamente donde están en la pantalla, incluso cuando la página se desplaza.

# Ejemplo Posicionamiento

```
<div id="navigation">
  <ul>
    <li><a href="this.html">This</a></li>
    <li><a href="that.html">That</a></li>
    <li><a href="theOther.html">The Other</a></li>
  </ul>
</div>
```

```
<div id="content">
  <h1>Ra ra banjo banjo</h1>
  <p>Welcome to the Ra ra banjo banjo page. Ra ra banjo
banjo. Ra ra banjo banjo. Ra ra banjo banjo.</p>
  <p>(Ra ra banjo banjo)</p>
</div>
```

```
#navigation {
  position: absolute;
  top: 0;
  left: 0;
  width: 200px;
}
```

```
#content {
  margin-left: 200px;
}
```

# Flotamiento (float)

- Al flotar una caja, la desplazará a la derecha o a la izquierda de una línea, con el contenido circundante fluyendo alrededor de ella.
- Normalmente, se utiliza el flotante para desplazar partes más pequeñas dentro de una página, como empujar un enlace de navegación a la derecha de un contenedor, pero también puede utilizarse con partes más grandes, como columnas de navegación.
- **static** es el valor predeterminado y representa un cuadro en el orden normal de las cosas, tal y como aparecen en el HTML.

```
#navigation {  
    float: left;  
    width: 200px;  
}
```

# Fuentes personalizadas

- Para embeber una fuente haremos uso de la directiva @font-face

```
@font-face {  
    font-family: "Nombre de fuente";  
    src: url(nombre_fuente.woff);  
}
```

- Luego usaremos la fuente con font-family

```
p {  
    font-family: "Nombre de fuente", arial, sans-serif;  
}
```

# Bordes redondeados y sombras

## ➤ Bordes redondeados

```
.caja_rond {  
    border-radius: 20px;  
}  
.caja_rond2{  
    border-radius: 6px 12px 18px 24px;  
}
```

## ➤ Sombras:

- box-shadow: horizontal-offset vertical-offset blur-radius spread-distance color

```
div {  
    box-shadow: 5px 5px 3px 1px #999  
}
```



## Herramientas de productividad

Revisa de estas herramientas de productividad

- [colorpicker.com](http://colorpicker.com)
- <https://fonts.googleapis.com/>
- <http://css3buttongenerator.com/>



# Pongámoslo en práctica

- Genera el HTML y CSS para la siguiente página
- [https://sdz-upload.s3.amazonaws.com/prod/upload/final\\_page.png](https://sdz-upload.s3.amazonaws.com/prod/upload/final_page.png)

The screenshot shows a travel diary website with the following structure:

- Header:** Zoror Travel diaries, with links to HOME, BLOG, RESUME, and CONTACT.
- Hero Image:** A large image of the Golden Gate Bridge with the caption "Reflections on my holiday in the United States..." and a "See article" button.
- Author Bio:** "I'M A GREAT TRAVELLER" followed by a short bio in Latin and a placeholder for a profile picture.
- About the Author:** A section featuring a cartoon zebra profile picture, the author's name (Zoror), birth date (23 November 2005), and a bio: "Let me introduce myself: My name's Zoror. I was born on 23 November 2005. A bit meager, is it not? This is why I now decided to write my biography to let my readers know who I really am."
- Social Media:** Icons for Facebook, Twitter, LinkedIn, and YouTube.
- Footer:** Sections for "MY LAST TWEET" (Hoy ha salido), "MY PICTURES" (with three thumbnail images), and "MY FRIENDS" (with a list of friends: Fafit the rabbit, Mr Bandub, Kavall, Prince, and Porceval.eu).

3

## Pseudo elementos

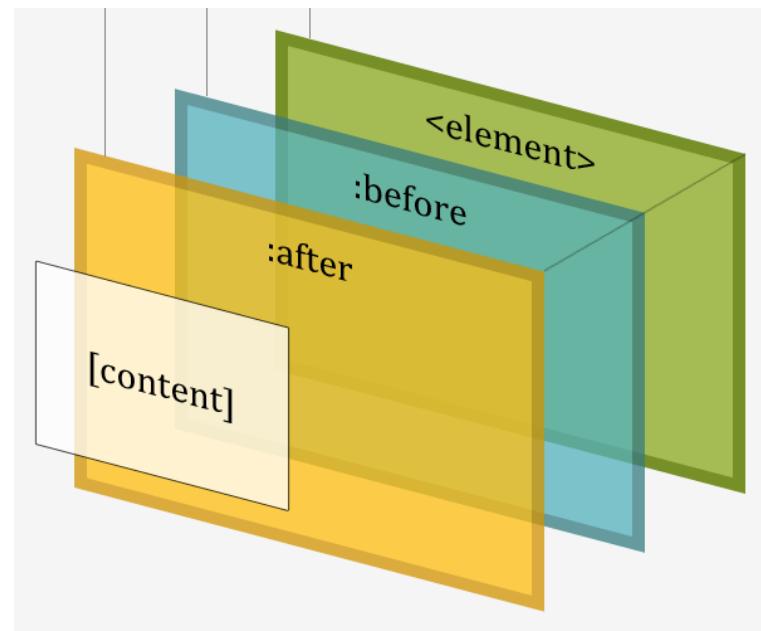
# Pseudo-Clases

- Son selectores que pueden actuar sobre información de elementos (podríamos decir que sobre su "estado")
- Esa información, se encuentra fuera del árbol HTML
  - Por ejemplo, los estados predefinidos de un elemento en función de su manejo por parte del usuario (cursor encima, enlace visitado, etc. )
- Ejemplo el elemento <a>
  - Estados normal (link), visitado (visited), cursor encima (hover) y pulsado (active)
  - Cada uno de esos estados podemos definirlo independientemente
  - Su sintaxis consiste en anexar el pseudo-elemento al nombre de la etiqueta mediante el símbolo de (:), como por ejemplo en

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

# Modelo de caja con pseudo-elementos

- Los pseudo-elementos :before y :after se construyen como capas adicionales que pueden añadirse a un elemento cualquiera.
- Cada capa puede contener gráficos y texto, así como algunos símbolos especiales: circle, disc, square, etc.
- Múltiples posibilidades para completar la información de cualquier elemento.



```
.contenedor div:after {  
    content:url(Graficos/HTML5_sintexto.png);  
}
```

# Pseudo-Clases y Pseudo-Elementos

Selector	Ejemplo	Descripción
<a href="#">:active</a>	a:active	Selects the active link
<a href="#">:checked</a>	input:checked	Selects every checked <input> element
<a href="#">:disabled</a>	input:disabled	Selects every disabled <input> element
<a href="#">:empty</a>	p:empty	Selects every <p> element that has no children
<a href="#">:enabled</a>	input:enabled	Selects every enabled <input> element
<a href="#">:first-child</a>	p:first-child	Selects every <p> elements that is the first child of its parent
<a href="#">:first-of-type</a>	p:first-of-type	Selects every <p> element that is the first <p> element of its parent
<a href="#">:focus</a>	input:focus	Selects the <input> element that has focus
<a href="#">:hover</a>	a:hover	Selects links on mouse over

# Pseudo-Clases y Pseudo-Elementos

Selector	Ejemplo	Descripción
<a href="#">:in-range</a>	input:in-range	Selects <input> elements with a value within a specified range
<a href="#">:invalid</a>	input:invalid	Selects all <input> elements with an invalid value
<a href="#">:lang(language)</a>	p:lang(it)	Selects every <p> element with a lang attribute value starting with "it"
<a href="#">:last-child</a>	p:last-child	Selects every <p> elements that is the last child of its parent
<a href="#">:last-of-type</a>	p:last-of-type	Selects every <p> element that is the last <p> element of its parent
<a href="#">:link</a>	a:link	Selects all unvisited links
<a href="#">:not(selector)</a>	:not(p)	Selects every element that is not a <p> element
<a href="#">:nth-child(n)</a>	p:nth-child(2)	Selects every <p> element that is the second child of its parent
<a href="#">:nth-last-child(n)</a>	p:nth-last-child(2)	Selects every <p> element that is the second child of its parent, counting from the last child

# Pseudo-Clases y Pseudo-Elementos

Selector	Ejemplo	Descripción
<a href="#">:nth-last-of-type(n)</a>	p:nth-last-of-type(2)	Selects every <p> element that is the second <p> element of its parent, counting from the last child
<a href="#">:nth-of-type(n)</a>	p:nth-of-type(2)	Selects every <p> element that is the second <p> element of its parent
<a href="#">:only-of-type</a>	p:only-of-type	Selects every <p> element that is the only <p> element of its parent
<a href="#">:only-child</a>	p:only-child	Selects every <p> element that is the only child of its parent
<a href="#">:optional</a>	input:optional	Selects <input> elements with no "required" attribute
<a href="#">:out-of-range</a>	input:out-of-range	Selects <input> elements with a value outside a specified range
<a href="#">:read-only</a>	input:read-only	Selects <input> elements with a "readonly" attribute specified
<a href="#">:read-write</a>	input:read-write	Selects <input> elements with no "readonly" attribute
<a href="#">:required</a>	input:required	Selects <input> elements with a "required" attribute specified
<a href="#">:root</a>	root	Selects the document's root element
<a href="#">:target</a>	#news:target	Selects the current active #news element (clicked on a URL containing that anchor name)
<a href="#">:valid</a>	input:valid	Selects all <input> elements with a valid value
<a href="#">:visited</a>	a:visited	Selects all visited links

# Ejemplos

```
p:not(.MiClase) {  
    color: red;  
    font-style: italic;  
}
```

```
<div>  
    <p>Párrafo 1</p>  
    <p>Párrafo 2</p>  
    <p>Párrafo 3</p>  
    <p>Párrafo 4</p>  
</div>
```

```
p:nth-of-type(3){  
    font-size:18px;  
    color: Red;  
}
```

4

## Más sobre CSS

# Barras de scroll

- La aparición o no de barras de scroll se puede controlar con la propiedad **overflow**
  - overflow: auto → deja al navegador decidir si muestra o no el scroll
  - overflow: scroll → fuerza muestrar el scroll
  - overflow: hidden → fuerza ocultar el scroll
- Se puede especificar para el scroll vertical u horizontal independientemente
  - Overflow-x: auto → deja al navegador decidir si muestra o no el scroll
  - Overflow-y: scroll → fuerza muestrar el scroll
  - Overflow-x: hidden → fuerza ocultar el scroll

# CSS Tables

- Permiten establecer una disposición visual similar al de las tablas HTML, pero sin las etiquetas HTML correspondientes
- Están soportados por todos los navegadores.
- Su uso se basa en los valores **table** y **table-cell** de la propiedad **display**

```
.parent {  
    display: table;  
    /*width: 200px;*/  
}  
  
.parent div{  
    display: table-cell;  
    text-align: center;  
    vertical-align:middle;  
}
```

# CSS Tables

## ➤ Equivalencias

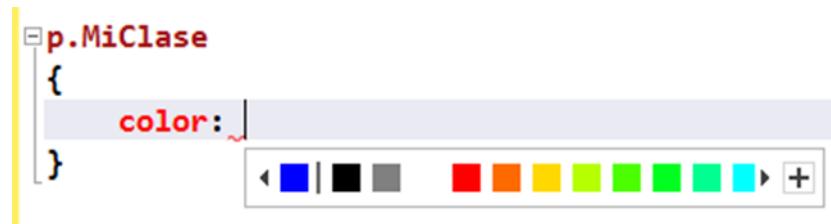
```
table { display: table }
tr { display: table-row }
thead { display: table-header-group }
tbody { display: table-row-group }
tfoot { display: table-footer-group }
col { display: table-column }
colgroup { display: table-column-group }
td, th { display: table-cell }
caption { display: table-caption }
```

# Más sobre los selectores (revisión)

## ➤ Selectores dependientes

Viene definido en función de otro selector ya existente

## ➤ Por ejemplo:



- Podemos usar varias clases simultáneamente en un elemento HTML indicándolo en el atributo class y separando los nombres de las clases con espacios

## ➤ El selector universal

Se define por el signo \*

```
* {  
    margin: 3px;  
    padding: 5px;  
}
```

# Más sobre los selectores

- Se puede utilizar una sintaxis específica para indicar el tipo de relación jerárquica que se desea establecer, de acuerdo con lo que vemos en la tabla siguiente:

Formato	Selector	Condición
<b>a b c</b>	Descendiente	<u>c</u> descendiente de <u>b</u> descendiente de <u>a</u>
<b>a * b</b>	Universal	<u>b</u> dentro de <u>a</u> para cualquier ancestro de <u>b</u>
<b>a &gt; b</b>	Hijo directo	<u>b</u> es hijo directo de <u>a</u>
<b>a + b</b>	Adyacente del mismo nivel	<u>b</u> es adyacente a <u>a</u> y de su mismo nivel
<b>a ~ b</b>	Mismo nivel	<u>b</u> es del mismo nivel que <u>a</u>

# Más sobre los selectores

- Por ejemplo, con la siguiente sintaxis:

```
.MiClase > div > p { color:green }
```

- ...estamos indicando que todos los elementos <p> que sean hijos directos de un elemento <div> cuyo ancestro tenga el atributo class con un valor "MiClase", irán en color verde.
- Mediante estas combinaciones de operadores y selectores, siempre podemos encontrar una opción de selección por especial que sea.
  - Por posición (relativa o absoluta) en el DOM
  - Por identificación (los identificadores deben ser únicos por definición.)
  - Por clase
  - Por una combinación cualquier de esos factores

# Más sobre los selectores (revisión)

## ➤ Valores enumerados

- La agrupación de valores, indicando una lista enumerada, separada por comas, tiene sentido en ciertas propiedades para indicar alternativas

font-family: Arial, Tahoma, 'Segoe UI', 'Times New Roman';

## ➤ Selectores por valor de atributos

- En CSS 2 se introdujo la noción de selectores por valor de atributos
- Son formas de seleccionar elementos por los valores que poseen alguno de sus atributos
- Lista de definiciones sintácticas

Elemento [attr] {} /\* Selector de atributo simple \*/

Elemento [attr='value']{} /\* Selector de comparación atributo/valor\*/

Elemento [attr~=value']{} /\* Selector de comparación parcial \*/

Elemento [attr|=value']{} /\* Selector de atributo por lenguaje \*/

# Más sobre los selectores (revisión)

- Imaginemos que tenemos las siguientes 4 etiquetas como parte de un menú:

```
<a href="Page1.html" rel="Value11" lang="en-GB">Link 1</a>
<a href="Page2.html" rel="Value11 2" lang="en-US">Link 2</a>
<a href="Page3.html" rel="Value13" lang="en-AU">Link 3</a>
<a href="Page4.html" rel="Value14" lang="es-ES">Link 4</a>
```

- El siguiente código CSS selecciona los 4 elementos link y los pone todos en rojo:

```
a[rel] { color: red; }
```

- Los que, teniendo el atributo, coincidan con el valor:

```
a[rel='Value1'] { color: green; }
```

# Más sobre los selectores (revisión)

- Elementos cuyo valor coincide parcialmente con la cadena indicada:

```
a[rel~='Value11'] { color: #b200ff; }
```

- Aquellos elementos que indiquen su atributo lang (lenguaje):

```
a[lang|='es'] { color: #cdb281; }
```

- Como vemos, el lenguaje se ha seleccionado de forma parcial, por lo que afectaría a todos los elementos cuyo lenguaje fuera una variante del español (es-\*).

5

# CSS3 Dinámico

# Transformaciones

- La especificación relativa a transformaciones se parece bastante, en cuanto al propósito a sus equivalentes de estas herramientas
- existen dos especificaciones separadas para trabajar con las transformaciones
  - Dos dimensiones: <http://www.w3.org/TR/css3-2d-transforms/>
  - Tres dimensiones: <http://www.w3.org/TR/css3-3d-transforms/>

# Propiedades de transformaciones

- En la siguiente tabla se muestra una lista

Propiedad	Descripción
<a href="#"><b>transform</b></a>	Aplica una transformación 2D o 3D a un elemento
<a href="#"><b>transform-origin</b></a>	Establece el punto de origen para las transformaciones
<a href="#"><b>transform-style</b></a>	Indica cómo se disponen en un espacio 3D elementos anidados
<a href="#"><b>perspective</b></a>	Especifica la perspectiva para los elementos 3D
<a href="#"><b>perspective-origin</b></a>	Indica la posición inferior de elementos 3D
<a href="#"><b>backface-visibility</b></a>	Define si un elemento debe ser visible o no cuando se orienta hacia la pantalla.

# Transformaciones

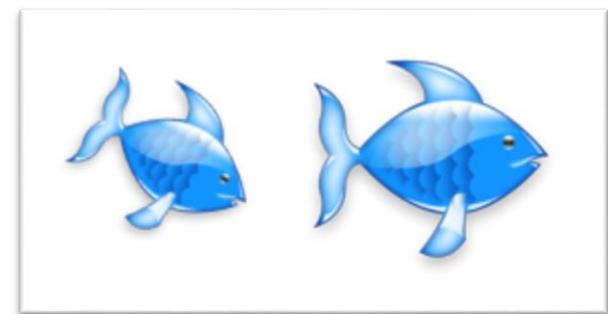
- La lista de valores posibles para las transformaciones aparece en la Tabla:

Propiedad	Descripción
rotate(<angle>)	Rota un ángulo
rotateX(<angle>)	Rota un ángulo en el eje de las X
rotateY(<angle>)	Rota un ángulo en el eje de las Y
scale(<numx2>,)	Escalado
scaleX(<number>)	Escalado en el eje de las X
scaleY(<number>)	Escalado en el eje de las Y
skew(<anglex2>,)	Deformación o Elongación para un ángulo
skewX(<angle>)	Deformación en el eje de las X
skewY(<angle>)	Deformación en el eje de las Y
translate(<lengthx2>,)	Desplazamiento (en el plano X/Y)
translateX(<length>)	Desplazamiento en el eje de las X
translateY(<length>)	Desplazamiento en el eje de las Y
matrix(<variousx6>,)	Matriz de transformaciones 3x3 capaz de representar cualquiera de las anteriores individualmente o de forma combinada

# Transformaciones

- Respecto al significado de los números, un valor positivo desplaza hacia la derecha o en el sentido de las agujas del reloj, mientras que uno negativo lo hace en sentido contrario.
- Lo recomendable en este momento es utilizar las extensiones de navegadores para hacer las pruebas de funcionamiento
- Por ejemplo, el siguiente código sirve para FireFox, Chrome, Safari y Opera:

```
#Fish1{  
    -moz-transform: scale(.75) rotate(30deg);  
    -webkit-transform: scale(.75) rotate(30deg);  
    -o-transform: scale(.75) rotate(30deg);  
    transform: scale(.75) rotate(30deg);  
}
```



# Transformaciones

- Podemos combinar los valores indicados en la Tabla de transformaciones indicando cada una de ellas separada por un espacio
- En algunos casos, como el de las transformaciones de rotación en uno de los ejes, es preciso indicarle esta circunstancia al elemento contenedor.
- Por ejemplo, utilizamos una imagen de un CD para apreciar mejor esta proyección espacial y escalamos el tamaño del segundo dibujo, rotándolo en el eje vertical (la proyección en 2D realmente) de la imagen, con un código como el siguiente:

```
#CD2
{
    -moz-transform: scale(1.5) rotateY(60deg);
    -webkit-transform: scale(1.5) rotateY(60deg);
    transform: scale(1.5) rotateY(60deg);
}
```



# Transiciones

- Una transición, como lo define el documento oficial de la W3C, “permite que los cambios de los valores en propiedades CSS se produzcan a lo largo de un período de tiempo predeterminado”.
- Aunque también puede suceder en tiempo 0 (instantáneamente), como cuando establecemos una condición inicial y una situación que cambia esa condición:

```
#Fish:hover { opacity:1; }  
#Fish { opacity:.333; }
```

# Transiciones

- Para añadir dinamismo al proceso, haremos que se realice a lo largo del tiempo, estableciendo un parámetro transition que admite 3 tipos de valores posibles:
  - El tipo de transformación sobre la que aplica el cambio temporal (all significa cualquiera, en este caso)
  - El tiempo que queremos que dure el proceso (medido en segundos o en milisegundos, debiendo indicar la unidad de medida)
  - La función de easing (los conocedores de Silverlight, reconocerán estos valores inmediatamente), que establece si es uniforme o no el proceso durante el tiempo establecido, con lo que se obtienen diversos efectos.

# Transiciones

- Por ejemplo, este mismo proceso, podemos escribirlo añadiendo la siguiente transición:

```
#Fish:hover {  
    opacity:1;  
    -webkit-transition: all 1000ms ease-in;  
}  
}
```

- Si queremos generalizarlo para el resto de navegadores, ya sabe el lector, la rutina es idéntica, pero con las extensiones:

```
#Fish:hover {  
    opacity:1;  
    -webkit-transition: all 1000ms ease-in;  
    -moz-transition: all 1000ms ease-in;  
    -ms-transition: all 1000ms ease-in;  
    -o-transition: all 1000ms ease-in;  
}  
}
```

# Transiciones

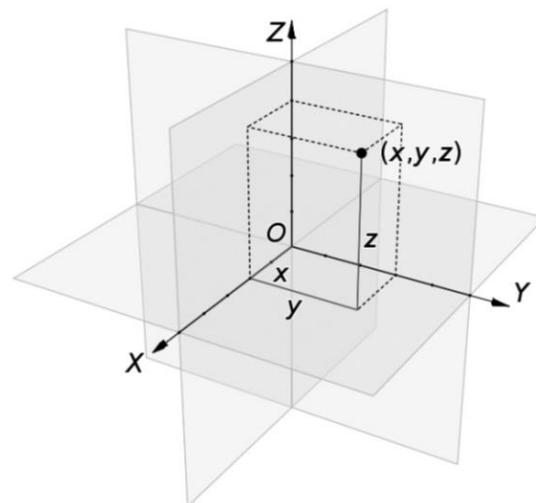
- Una vez terminada la transición, podemos añadir funcionalidad suscribiéndonos al evento transitionEnd (o podemos realizar una acción previa a la animación manejando el evento complementario transitionStart)

```
var div1 = document.getElementById("div1");
div1.addEventListener("transitionend", onTransitionend, true);

function onTransitionend(e) {
    var nameOfProperty = e.propertyName;
    var elapsedTime = e.elapsedTime;
    alert(nameOfProperty + " - " + elapsedTime);
}
```

# Transiciones en 3D

- Lo primero que tenemos que tener en cuenta al trabajar en entornos 3D es que la adición de una dimensión tiene efectos notables sobre la programación
- El soporte por el momento sigue creciendo, pero se recomienda (en caso de utilizar uno de estos recursos), añadir todas las extensiones disponibles más la estándar
- En 3D el sistema de coordenadas funciona basándose en lo que vemos en la figura



# Transiciones en 3D

- Cualquier punto queda definido por sus 3 coordenadas a los ejes X, Y, Z
- La parte positiva del eje de las Z apunta al espectador
- Este es el conjunto completo de transformaciones y efectos aplicables a los contextos 3D

Valor	Descripción
<b>perspective(&lt;número&gt;)</b>	Define la profundidad. A valores más bajos, mayor profundidad
<b>rotate3d(&lt;número x3&gt;,&lt;ángulo&gt;)</b>	Efectúa una rotación en el punto 3D indicado por x,y,z, en un ángulo establecido por el 4º parámetro. Los valores positivos realizan la rotación en el sentido de las agujas del reloj, y los negativos en sentido contrario.
<b>rotateX(&lt;ángulo &gt;)</b>	Gira un ángulo indicado solo en el eje X
<b>rotateY(&lt;ángulo &gt;)</b>	Gira un ángulo indicado solo en el eje Y
<b>rotateZ(&lt;ángulo &gt;)</b>	Gira un ángulo indicado solo en el eje Z
<b>translate3d(&lt;longitudx3&gt;,)</b>	Desplaza el elemento en el espacio en los valores indicados para cada parámetro (los positivos acercan, y los negativos, alejan)
<b>translateX(&lt;longitud &gt;)</b>	Idéntico al anterior, pero solo para el eje de las X

# Transiciones en 3D

- La unión de las transiciones y las capacidades de animación en 3D, permitirán suplir con creces los efectos que se conseguían hasta ahora mediante complementos adicionales como Flash o Silverlight

Valor	Descripción
<b>translateY(&lt;longitud &gt;)</b>	Idéntico al anterior, pero solo para el eje de las Y
<b>translateZ(&lt;longitud &gt;)</b>	Idéntico al anterior, pero solo para el eje de las Z
<b>scale3d(&lt;número ×3&gt;,)</b>	Escala un elemento en las 3 dimensiones tomando el valor factor de multiplicación. Los valores negativos no solo disminuyen el tamaño, sino que reflejan el elemento a lo largo del eje.
<b>scaleX(&lt;número &gt;)</b>	Funciona como el anterior, pero aplicándolo solamente al eje X
<b>scaleY(&lt;número &gt;)</b>	Funciona como el anterior, pero aplicándolo solamente al eje Y
<b>scaleZ(&lt;número &gt;)</b>	Funciona como el anterior, pero aplicándolo solamente al eje Z
<b>matrix3d(&lt;varios×16&gt;)</b>	Matriz de 4x4 = 16 valores usada para establecer un conjunto de trasformaciones en una sola operación.

# Transiciones en 3D

- Es de esperar que, en breve exista soporte de esta característica en más navegadores, ya que, de momento, es muy precaria.
- Los ejemplos, seguirían los patrones implementados en el caso de 2D.

# Animaciones

- Una animación es una definición de interpolación que no se asocia a ningún elemento concreto.
- Posteriormente, podemos utilizarla con los elementos que queramos, favoreciendo al idea de reutilización del código.
- Se define mediante la directiva @Keyframes
- Al menos, requiere de los 2 puntos de interpolación básicos (origen y destino), expresados mediante from y to:

```
@keyframes Animacion1 {  
    from {  
        left: 100px;  
        right: 50px;  
    }  
    to {  
        left: 10px;  
        right: 10px;  
    }  
}
```

# Animaciones

- Con esta definición, podemos aplicar la animación estableciéndola para cualquier selector, indicando:
  - La duración prevista
  - El número de propiedades cambiantes que queremos animar
  - El tiempo de retardo inicial
  - La curva "easing" que queremos aplicar y su dirección
  - El numero de veces que queremos que se repita

```
#Dibujo1:hover{  
    animation-name: Animacion1;  
    animation-duration: 3s;  
    animation-timing-function: ease-out;  
    animation-delay: .1s;  
    animation-iteration-count: 2;  
    animation-direction: normal;  
}
```

# Animaciones

- Por supuesto, tendremos que utilizar la sintaxis con prefijos de vendedor, si queremos que sea reconocido por todos los navegadores

```
#Dibujo1:hover {  
    -moz-animation-name: Animacion1;  
    -o-animation-name: Animacion1;  
    -webkit-animation-name: Animacion1;  
    animation-name: Animacion1;  
}
```

- El motor de CSS creará una hilo de ejecución por cada animación y eventualmente, nos indicará tanto su puesta en marcha como su conclusión a través de 2 eventos:
  - animationStart
  - animationEnd

# Animaciones

- Estos eventos se programan de forma similar a como hemos visto en las transiciones.
- Dado que una animación puede llevar un número de iteraciones, también podemos usar el evento animationIteration
- La tabla siguiente muestra los prefijos de vendedor necesarios según el navegador:

W3C standard/ Firefox	webkit	Opera	IE10
animationstart	webkitAnimationStart	oanimationstart	MSAnimationStart
animationiteration	webkitAnimationIteration	oanimationiteration	MSAnimationIteration
animationend	webkitAnimationEnd	oanimationend	MSAnimationEnd

# Animaciones

- Para que su programación sea independiente del prefijo utilizado, podemos usar la técnica de declarar los eventos para todos sin excepción:

```
// aplicar manejadores de evento
pfx = ["webkit", "moz", "MS", "o", ""];
function PrefixedEvent(element, type, callback) {
    for (var p = 0; p < pfx.length; p++) {
        if (!pfx[p]) type = type.toLowerCase();
        element.addEventListener(pfx[p] + type, callback, false);
    }
}

// eventos de animación
PrefixedEvent(anim, "AnimationStart", AnimationListener);
PrefixedEvent(anim, "AnimationIteration", AnimationListener);
PrefixedEvent(anim, "AnimationEnd", AnimationListener);
```



## Pongámoslo en práctica

- Añade animaciones a tu página al mostrarse las imágenes



## Dándole personalidad al prototipo

- En equipo, añade los elementos gráficos al prototipo HTML de BananaTube



[...]**netmind**

WeKnowIT

Barcelona

C. Almogàvers, 123  
08018 Barcelona  
Tel. 93 304.17.20  
Fax. 93 304.17.22

Madrid

Plaza Carlos Trías Bertrán, 7  
28020 Madrid  
Tel. 91 442.77.03  
Fax. 91 442.77.07

[www.netmind.es](http://www.netmind.es)



GOBIERNO  
DE ESPAÑA

MINISTERIO  
DE ENERGÍA, TURISMO  
Y AGENDA DIGITAL

**red.es**



ESTRATEGIA DE  
EMPRENDIMIENTO Y  
EMPLEO JUVENIL  
*garantía juvenil*



Agenda Digital para España



**UNIÓN EUROPEA**

Fondo Social Europeo  
*“El FSE invierte en tu futuro”*