APUNTES HTML5: CSS

José Juan Urrutia Milán

Reseñas

- Curso HTML5: https://www.youtube.com/playlist?list=PLU8oAlHdN5BnX63lyAeV0LzLnpGudgRrK
- Curso CSS: https://www.youtube.com/playlist?list=PLU8oAlHdN5BmpUDdnWSglIIHfIosElaVN
- w3c (página que decide los estándares CSS/HTML...): www.w3c.org

Siglas/Vocabulario

- **IDE:** Entorno de Desarrollo Integrado.
-

Leyenda

Cualquier abreviatura o referencia será subrayada.
Cualquier ejemplo será escrito en **negrita.**
Cualquier palabra de la que se pueda prescindir irá escrita en cursiva.
Cualquier abreviatura viene explicada a continuación:

Abreviaturas/Referencias:

- 123: Hace referencia a cualquier número.
- nombre: Hace referencia a cualquier palabra/cadena de caracteres.
- a: Hace referencia a cualquier caracter.
- cosa: Hace referencia a cualquier número/palabra/cadena.
- Tipo_var o **…** : Hace referencia a cualquier tipo de variable primitiva o de tipo String.
- nombre_var: Hace referencia a cualquier nombre que se le puede dar a una variable.
- código: Hace referencia a cualquier instrucción. (Se usará para indicar dónde se podrá inscribir código.)
- variable: Hace referencia a cualquier variable.
- condición: Hace referencia a cualquier condición. Entiéndase por condición, una afirmación que devuelve un true o un false. Ej: (variable == 123). *Una variable del tipo boolean puede ser usada como una condición.

Índice

text-align

color

font-family

font-size

font-weight

font

text-decoration

text-transform

background-color

background

background-image

background-repeat

background-position

opacity

height

width

margin-left/right/top/bottom

margin

padding-left/right/top/bottom

padding

display

align-items

justify-content

place-items

float

border-right/left/top/bottom

border-right/left/top/bottom-color

border

border-color

border-radius

border-collapse

outline

outline-offset

box-shadow

text-shadow

cursor

line-height

letter-spacing

overflow

top/left/right/bottom

list-style

visibility

position

Título I: Importar fuentes propias

Título II: Degradados

Tipos de degradado

linear

radial

Especificar navegador

Porcentaje de cada color

Título III: Colores

rgb(r, g, b)

rgba(r, g, b, a)

Título IV: Transformaciones

Escalar componentes

Invertir componente

Rotar componente

Cambiar la perspectiva

Animaciones

Título V: Medidas

Porcentaje

Píxeles

em

Introducción

El objetivo de este curso es aprender CSS y su implementación en HTML.
CSS es un lenguaje de diseño gráfico usado sobre todo para dotar de estilo a algunos lenguajes estructurales o de etiquetas como html.
CSS funciona como hojas de estilo en cascada.
El código CSS de una página HTML se puede indicar en 3 sitios diferentes:

1. En un archivo independiente con extensión .css.
2. Dentro de la etiqueta head de un documento html.
3. Dentro de una etiqueta inline en casos concretos (desaconsejada).

Capítulo I: Conceptos básicos

Comentarios

Para realizar comentarios en CSS (partes que no se tendrán en cuenta), debemos especificar /* antes y */ después:

```
/*p{
        color:#00F
}*/
```

Título I: Estilo

Un estilo CSS es una regla que le indica al navegador cómo aplicar un formato a una página web. Este estilo se suele indicar dentro de la etiqueta **<head>**, dentro de una nueva etiqueta **<style>**.
Cada navegador da distintas propiedades CSS por defecto.

*También podemos crear un archivo .css que importamos a nuestro archivo .html para usar los estilos (en cuyo caso, la primera línea de este archivo debe ser **@charset "utf-8";**).
También debemos indicar que vamos a usar esta hoja de estilos desde el documento HTML (Consultar Apuntes HTML, Capítulo I, Título II, Añadir documentos CSS).
**Si usamos ambos procedimientos, el <style> se impondrá sobre el archivo externo, de forma que si ambos modifican alguna parte del programa, sólo se mostrará los cambios del style.

Reseteo

A la hora de crear un estilo, puede sernos de ayuda resetear el estilo CSS por defecto que el navegador da a nuestro componentes. Esto lo hacemos con el código:

```
body, html, div, blockquote, img, label, p, h1, h2, h3, h4, h5, h6,
pre, ul, ol, li, dl, dt, dd, form, a, fieldset, input, th, td{
        margin: 0; padding: 0; border: 0; outline: none;
}

body{
        line-height: 1;
        font-size: 100%
}

h1, h2, h3, h4, h5, h6{
        font-size: 100%;
        padding: 0;
        margin: 0;
}
```

*Recomendable tener el código en un documento CSS desde el cual poder hacer un copy-paste.

Estilo inline

Podemos establecer que un único elementos tenga una propiedad en específico, esto lo hacemos con el atributo **style** de las etiquetas html:
** texto **

Partes de un estilo

Un estilo tiene dos partes, un selector (que hace referencia a las zonas de una página web que queremos modificar) y unas instrucciones o bloque (que indican el formato o propiedad a aplicar).

Aplicar estilo

Dentro de la etiqueta style, primero indicamos el selector (en este caso, queremos modificar una etiqueta h1), por lo que indicamos **h1** y

abrimos un bloque de declaración (con llaves) donde indicamos un bloque con el formato clave:valor:
*Separamos las diferentes propiedades con ; entre sí.

h1{text-align:center; color:#00F}

**Con el código superior, le estamos aplicando un estilo a todas las etiquetas h1 de nuestra página.
***Indicar **body** como selector cuando queramos cambiar alguna propiedad del body en sí (como el color de fondo).
****Un selector puede tomar como referencia una etiqueta estructural, como un header: **header{background-color:"00F"}**

Título II: Tipos de selectores

Existen diferentes tipos de selectores: de etiqueta (los que hemos usado), de id y de clase.
Separar selectores intercalando comas para aplicar un mismo estilo a diferentes selectores. Esto es llamado selector de grupo:
h1, h2, h3{<u>código</u>}

Selector de etiqueta

Especificamos el nombre de la etiqueta y el estilo especificado se aplica a todas las etiquetas de este tipo.
*Combiar con selector descendiente.

Selector descendiente

Podemos seleccionar las etiquetas de un tipo especificado que sean hijas de otro tipo especificado:
figure img{}
Así, sólo recibirán este estilo todas las etiquetas **img** que se encuentren dentro de un **figure** .
*Podemos usar los selectores descendientes con cualquier selector.

**Podemos usar tantos selectores descendientes como queramos:

#principal .componentes p strong{código}
Dará formato a todas las etiquetas **strong** que estén dentro de un **p** que esté dentro de un componente de la clase **componentes** y de un componentes con id **principal** .

***Los selectores descendientes son de los más usados.
****Los selectores descendientes se pueden saltar componentes:
(HTML):
<ul id=”lista”>
Pínchame

(CSS):
#lista a{código}
De esta forma, aplicamos los estilos especificados a todos los **<a>** que estén dentro del componente con id = lista, sin nombrar la etiqueta **<li>**.
*****A la hora de indicar los componentes de un selector descendiente, se puede indicar un asterisco para indicar que los estilos se aplicarán a cualquier etiqueta que esté dentro de un componente especificado:
#principal *{código}
Se aplicará a todas las etiquetas que estén dentro de un componente con el id = principal.

id

Los selectores de id nos permiten establecer estilos a componentes específicos de nuestro documento html.
Por ejemplo, queremos establecer un color de fondo a un header pero no a todos los header que tenemos, sino a uno en concreto, por lo que usamos selectores de id:

(CSS)
#cabecera{background-color:”#F00”}

(HTML)
<header id=”cabecera”>...</header>

id cuando se pone el ratón encima

Podemos especificar el estilo que adquirirá un componente cuando se le ponga el ratón encima y que este vuelva a su estado default cuando se quite el ratón, para esto, creamos una nueva etiqueta donde especificamos las modificaciones que sufrirá cuando se sitúe el ratón encima, especificando **:hover** :
#cabecera:hover{background-color:”#0F0”}

De esta forma, se mostrará un objeto con el color F00, cuando se sitúe el ratón encima, cambiará a 0F0 y cuando se quite, F00.
*Especificar **focus** en vez de **over** para aplicar cuando el componente tenga el foco.
**Verdaderamente, este es un selector mediante pseudoclases.
***Este selector lo podemos usar con otros que no sean de id, como de etiqueta.

clase

Podemos especificar una clase a etiquetas html para poder acceder a ellas desde css:
(CSS)
.importante{color:#00F}

(HTML)
** Hola **

*También podemos especificar un selector para sólo las etiquetas de un tipo que estén dentro de una clase:

p.importante{código}
El estilo sólo será para aquellas etiquetas **p** que estén dentro de la clase **importante**. Los demás elementos que no sean p y estén dentro de **importante**, ignorarán esta instrucción.

name

Podemos establecer un **name="txt"** a modo de "id" (aunque un mismo name se puede especificar en diferentes elementos, no como el id) y desde código CSS, dar un estilo específico a todas las etiquetas de una clase que tengan el mismo **name**:
(CSS)
p[name="elementos"]{color:#F00}

(HTML)
** Hola **
** Cómo estás **
** Adios **

De esta forma, sólo tendrán el color especificado la primera y la última etiqueta.

-Caracteres especiales: Podemos añadir tres caracteres especiales a un name de forma que css aplique la propiedad a name que empiecen como el texto, que terminen como el texto o que contengan las letras especificadas:
[name^="txt"] Empieza por txt.
[name$="txt"] Termina por txt.
[name*="txt"] Contiene txt.
*Este selector además de con **name**, se puede aplicar con cualquier propiedad **html** (como un **href** de un **<a>)** .

pseudoclases

Se pueden aplicar propiedades a ciertos elementos que forman parte de una pseudoclase (*).
*Una pseudoclase es un grupo de elementos hermanos que se encuentran dentro de una etiqueta superior.
**Las peseudoclases se pueden combinar con otros tipos de selectores, como selectores descendientes o de id.

Para ello:
(CSS)
p:nth-child(2){color:#F00}
En vez de un número, se puede introducir **odd** o **even**.

(HTML)
** Hola **
** Cómo estás **
** Adios **
De esta forma, se aplicará el color especificado a la segunda etiqueta.

*Existen variantes de la instrucción **nth-child(n):**
first-child, **last-child**, **only-child** (para hijos únicos).
Los vínculos o etiquetas **a tienen valores especiales para las pseudoclases. (ver Apuntes HTML, Capítulo II, Título XIII, Estados)

selectores con pseudo elementos

Permite seleccionar pseudo elementos (los cuales son porciones de un elemento que fácilmente podrían ser otro elemento).
Para ello, especificar seleccionador, dos puntos y el espacio en el que está el pseudo elemento. Por ejemplo, para dar color azúl a la primera línea de todos los párrafos:

```
p:first-line{
    color:#00F;
}
```

*Esto también se podría hacer creando una etiqueta **span** en el archivo HTML con una id que identificaríamos en CSS.

selectores hermano

Permite seleccionar todos los componentes de un tipo que estén inmediatamente después de un componente de otro tipo:
h1+p{código}
De esta forma, sólo los p que estén detrás de un h1 obtendrá este estilo.

Título III: Herencia de estilos

En CSS, podemos hacer que componentes hijo hereden algunos estilos de componentes padre. Para ello, implementar propiedades en el componente padre y el hijo, generalmente, las heredará.
Por todo esto, si aplicamos un estilo al **body** , todos los componentes debería heredar las propiedades del **body**.
Por ejemplo, las etiquetas **** dentro de etiquetas **<p>** heredan la propiedad **color** de **<p>**.
La herencia funciona también con estilos de clase o de id.

*Cuando un componente recibe dos estilos, uno por herencia y otro específico, el más específico es dominante entre ambos (ver Título IV).

Excepción

Hay algunas propiedades que dependiendo de a qué elemento se le aplique puede que no se hereden. Por ejemplo: las etiquetas h**x** de títulos no heredan tamaños de letra, sino que los ignoran y se quedan con el tamaño por defecto correspondiente por cada **h**x.

Esto sucede porque algunos elemento HTML ya llevan incorporados algunos estilos específicos, por lo que eligen estos antes que los heredados.

Título IV: Cascada

La cascada son reglas que determinan qué propiedades de estilo se aplican a un elemento en concreto cuando existe en él un conflicto de estilos.

Conflicto de estilos

Existe un conflicto de estilos cuando entra en juego la herencia o cuando se aplican varios estilos al mismo elemento.
También cuando los componentes HTML ya llevan especificados ciertos estilos (como etiquetas a).

Prioridad

La regla es coger el estilo más específico:

inline > id > clase > etiqueta.

Para orientarnos, es como si cada estilo tuviese diferentes puntos de importancia:

- **etiqueta :** 1 pto.
- **clase :** 10 ptos.
- **id :** 100 ptos.
- **inline :** 1000 ptos.

En selectores descendientes, estas importancias se suman, por lo que si aparece un id y una etiqueta, es como si valiera 101 ptos.
Por lo que si jugamos con estos puntos, podemos establecer cuál será el estilo más específico.
*Los estilos inline no son recomendables.

Capítulo II: Propiedades

(Consultar propiedades CSS en Google).
(ver Título V).
*Cuando se indique que una medida es en grados, especificar medida y luego **deg**, según: **ndeg**
****Cuando** se indique que una medida es en segundos, especificar medida y luego **s**, según: **ns**

Nomenclatura margin/padding

Un dato: Se guarda esa distancia por todas las direcciones.
Dos datos: Se guarda la primera distancia por arriba y por abajo y la segunda por la izquierda y la derecha.
Cuatro datos: Se guardan esas distancias por arriba, derecha, abajo e izquierda (en ese orden).

text-align

Indica el alineamiento del texto.

color

Indica el color del texto (Especificar almohadilla y tres dígitos hexadecimales (#00F) (o seis dígitos) o Consultar Título III).

font-family

Permite especificar el tipo de letra.
*Recomendable indicar varios tipos de letra por si la primera no está, se aplique la segunda.

font-size

Especifica el tamaño del texto en píxeles (o en porcentaje respecto al tamaño default).

font-weight

Establece el estilo del texto (**bold, normal** ...).

font

Especifica el estilo, tamaño y tipo de fuente del texto.
font: bold 36px Verdana, Geneva, sans-serif
*Se usará verdana primero, si no está instalada, geneva y si tampoco, sans-serif.
**Consultar título I: importar tipos de letra
***Especificar **normal** en estilo para que no sea ni negrita ni itálica.
****Si el tipo de letra contiene espacios, especificar entre “”:
font: bold 1em “Times New Roman”

text-decoration

Da un estilo especial (subrayado…).
Especificar **none** para quitar las decoraciones (como las por defecto de los vínculos).

text-transform

Pone todo el texto en mayúsculas (**uppercase**), minúsculas (**lowercase**) o la primera letra de cada palabra en mayúsculas (**capitalize**).

background-color

Indica el color de fondo (Especificar almohadilla y tres dígitos hexadecimales (#00F) o Consultar Título III).

background

Nos permite indicar un degradado como color de fondo.
Permite indicar en él cualquier valor de las propiedades **background-...** a la vez:
background: url(ruta) no-repeat center top;
(Consultar Título II: Degradados).

background-image

Nos permite indicar una foto como fondo:

background-image:url(foto.jpg);

*Si la imagen es más estrecha de lo necesario, esta se repetirá en bucle hasta ocupar toda la página. Para controlar esta repetición, usar característica **background-repeat**

**La raíz de la ruta es la carpeta de nuestro proyecto.

***En la mayoría de los casos, será necesario indicar un **padding**, **margin** o **background-position** para colocar el texto conforme a la imagen.

background-repeat

Permite controlar cómo se repite una imagen. Valores: (repeat, no-repeat, repeat-x, repeat-y ...).

background-position

Establece la posición de la foto de fondo especificando dónde se pondrá esta. Sigue la nomenclatura de margin y padding (ver Nomenclatura margin/padding).

background-position: center top;

*Se pueden especificar píxeles, los cuales se sumarán a la posición actual.

opacity

Establece la opacidad del componente, donde 1 es totalmente opaco y 0 totalmente transparente.

*Para especificar números <1, no hace falta especificar el 0:

componente{opacity:.5}

Interesante usarlo dentro de un selector **:hover .

height

Especifica el tamaño vertical en píxeles (o en porcentaje total del área que ocupa).

width

Especifica el tamaño horizontal en píxeles (o en porcentaje total del área que ocupa).

margin-left/right/top/bottom

Establece un margen en píxeles.
*Si el valor de margin es negativo, el siguiente componente se mete por encima de este componente.

margin

Establece la separación que dejará el componente por arriba y por abajo en píxeles y la separación por izquierda y derecha.
*Especificar **auto** para que se adapte a la resolución de la pantalla:
margin:15px auto o **margin: auto**
**Si el valor de margin es negativo, en vez de alejar el componente, lo acercaremos. Esto nos permite colocar algunos componentes encima de otros.

padding-left/right/top/bottom

Establece la separación del contenido a la etiqueta estructural (separación entre filo de <h1> y <header> , ...).

padding

Establece la separación del contenido al contenedor
Este se puede especificar en el contenido o en el contenedor.

display

Indica el tipo de etiqueta (**inline**, **block, flex, grid, …**)
(consultar HTML, Capítulo II, etiquetas inline AND etiquetas block).

align-items

Alinea los elementos verticalmente cuando display: flex; (align-items: center).

justify-content

Alinea los elementos horizontalmente cuando display: flex; (justify-content: center).

place-items

Alinea los elementos horizontalmente y verticalmente cuando display: grid; (place-items: center).

float

Indica la posición del elemento (left, right, top, bottom...), flotándolo, haciendo que su tamaño se adapte a su contenido y hace que el siguiente elemento le rodee.
*Especificar width (en %) para que no ocupe todo el ancho.
Para que el siguiente elemento no rodee al elemento flotante, agregar propiedad **clear:both al siguiente elemento.
***Esta propiedad (combinándola con **padding** , **margin** y **width**) nos permite crear información en barras laterales, como si tuviéramos nuestra página web dividida en columnas.

border-right/left/top/bottom

Establece un borde al componente por la parte especificada. Indicar grosor del borde en píxeles, tipo de borde y color del borde:
border:4px solid #00F
Tipos de borde: solid, dashed, dotted…
*Especificar **none** para eliminar el borde.

border-right/left/top/bottom-color

Establece el color del borde.

border

Establece un borde al componente. Indicar grosor del borde en píxeles, tipo de borde y color del borde: **border:4px solid #00F**
Tipos de borde: solid, dashed, dotted…
*Especificar **none** para eliminar el borde.

border-color

Establece el color del borde.

border-radius

Especifica en píxeles cómo de redondeados serán los bordes (consultar nomenclatura margin/padding).
*Recomendable especificar en algunos el navegador y luego uno genérico:
-moz- border-radius: 8px
-webkit-border-radius:8px
border-radius:8px

border-collapse

Especificar **collapse** para que los bordes de componentes que están muy juntos (como bordes de celdas de tablas) se junten en un único borde.
*Especificar en una etiqueta **table** .

outline

Permite crear un borde más exterior que **border** que envuelve al componente. Especificar ancho del borde en píxeles, tipo de borde y color:
outline:3px dashed #00F
Tipos de borde: solid, dashed, dotted...

outline-offset

Aleja el borde **outline** los píxeles especificados.

box-shadow

Establece el color, desplazamiento horizontal y desplazamiento vertical de la sombra de la caja en píxeles:

box-shadow:#999 5px 5px

También se puede añadir la difuminación de la sombra en píxeles para que parezca más real:

box-shadow:#999 5px 5px 10px

text-shadow

Establece el color. desplazamiento horizontal y desplazamiento vertical de la sombra del texto en píxeles:

text-shadow:#0F0 5px 5px

También se puede añadir la difuminación de la sombra en píxeles para que parezca más real:

text-shadow:#0F0 5px 5px 10px

*Se pueden añadir distintas sombras a la vez separándolas por una coma.

cursor

Establece que el cursor seleccione el componente como si de un link se tratase si se le indica el valor **pointer:**

cursor: pointer

line-height

Establece un interlineado entre las líneas.

letter-spacing

Establece una distancia horizontal en píxeles entre las letras.

overflow

Establece lo que hará un componente cuando su borde o componente externo choque con otro componente.

Especificar **hidden** para esconder el borde.

top/left/right/bottom

Deja los píxeles indicados de separación respecto a arriba (/izquierda/derecha/abajo)
*Se pueden especificar píxeles negativos para invertir la dirección.
Usar con valores absolute de **position.

visibility

Establece si el componente será visible (**"visible")** o no (**"hiden"**).

list-style

Aplicar sobre etiquetas **<li>**
Especifica el estilo de los puntos de las etiquetas.
Especificar **none** para que las etiquetas no tengan puntos.

position

Nos permite decir donde se posicionará el elemento. Acepta 4 valores:

- absolute : Saca el elemento del flujo HTML (ignora la estructura)
Nos permite colocar elementos "flotantes" encima de otros componentes, controlando su posición con las propiedades **left** y **top**.
El elemento sigue dentro de su componente padre.
Se utiliza sobre todo para reubicar elementos.

- relative : Saca el elemento del flujo HTML (ignora la estructura)
Nos permite colocar elementos "flotantes" encima de otros componentes, controlando su posición con las propiedades **left** y **top**.
El elemento sale fuera de su componente padre y deja un hueco donde debería estar. Si se anida algo en su interior, esto se coloca donde debería estar.
*Todas las medidas que se coloquen en él dejarán de hacer mención a la página y harán mención a él.
Especificar **clear:both para que este no se mueva de su sitio.

- fixed : Posicionamiento fijo e inalterable dentro del contenedor padre. Aunque nos desplacemos con barras de desplazamiento, el componente permanece fijo. Podemos controlar su posición con las propiedades **left** y **top**.

-static : Posicionamiento por defecto de las etiquetas HTML.

Título I: Importar fuentes propias

En algunas ocasiones, querremos usar tipos de letra raros que casi ningún ordenador tiene instalado y queremos que este se vea en todos los ordenadores que accedan a nuestra página web.
Para poder hacer esto, tenemos que importar nuestro tipo de letra de una forma especial en CSS (necesitamos tener nuestro tipo de letra en un archivo de formato .TTF):

```
@font-face{
        font-family: nombre;
        src: local (nombre);
        url: ('ruta');
}
```

Donde **nombre** es el nombre que le damos a nuestro tipo de letra (puede ser cualquiera, como si de una variable de un lenguaje de programación se tratara).
Y **ruta** es el archivo .ttf que se encuentra en la raíz de nuestro programa (si estamos en local, en la misma carpeta que nuestro .html o .css, si estamos en un servidor, en la raíz del servidor).
*Si en **ruta** se especifican directorios, es probable que dé error, por lo que se recomiendo que esté en la raíz.
**La fuente se descarga en el usuario de manera transparente. Si este ya la tiene, no se descargará.

Título II: Degradados

Para ello, debemos especificar el navegador, el tipo de degradado y los parámetros correspondientes a cada tipo.

Tipos de degradado

linear

Especificar como parámetros por dónde empieza el degradado (right, left, top o bottom), el primer color y el último color.
background:-webkit-linear-gradient(top, #00F, #3FF)

radial

Especificar como parámetros por donde empieza el degradado (center), tipo de degradado radial (**circle** , **ellipse**) y los colores de principio y fin.
background:-webkit-radial-gradient(center, circle, #00F, #3FF)

Especificar navegador

Esto es necesario para que el degradado se muestre correctamente en cada uno:

- **moz** para Firefox.
- **webkit** para Chrome.
- **ms** para Internet Explorer.
- **o** para Ópera.

*Especificar múltiples líneas iguales cambiando el navegador para que se muestre correctamente en todos.
background:-webkit-linear-gradient(top, #00F, #3FF)
background:-moz-linear-gradient(top, #00F, #3FF)...

Porcentaje de cada color

Se puede especificar en qué porcentaje aparecerá cada color:
background:-moz-linear-gradient(top, #00F 10%, #3FF 90%)
(Cambiar como guste).

*Este porcentaje se puede especificar en cualquier tipo de degradado.

Título III: Colores

Se pueden crear ciertos colores personalizados:

rgb(r, g, b)

Crea un color rgb a partir de los primarios especificados.

rgba(r, g, b, a)

Crea un color rgb a partir de los primarios especificados y le da la transparencia a.
*Donde **a** es un valor comprendido entre 0 y 1, donde 0 es transparente total y 1, opaco total.

Título IV: Transformaciones

Escalar componentes

Para ello, especificar navegador-**transform:scale(n)** , donde n específica la proporción que este aumenta o disminuye (0.5=mitad, 2=el doble, 1=la misma...)
(consultar Título II: Especificar navegador).
-webkit-transform:scale(0.5)
-o-transform:scale(2)
Así, reducimos a la mitad en Chrome y aumentamos al doble en Ópera.

Invertir componente

Para ello, indicar la escala y un -1:
-ms-transform:scale(2, -1)

Rotar componente

Indicar el navegador **-transform:rotate(n)** , donde n son los grados que este rota:

-webkit-transform:rotate(30deg)

Cambiar la perspectiva

Indicar el navegador-**transform:skew(n)** , donde n indica los grados de la perspectiva:
-o-transform:ske(10deg)

Animaciones

Podemos crear animaciones de la siguiente forma:
-navegador-transition:-navegador-transform tiempo tipo retardo
Donde **tiempo** es el tiempo que tarda en completarse la animación, **tipo** es el tipo de efecto (**ease, ease-in-out**) y **retardo** son los segundos de retardo en iniciarse.

-moz-transition:-moz-transform 1s ease 0.5s

*Los parámetros temporales pueden ser decimales.
Las animaciones se especifican dentro del selector "por defecto" (el que no lleva **:hover), y esta animación se ejecutará al poner el ratón encima y al quitarlo (terminando con el aspecto que se indicó en **:hover**).

Título V: Medidas

Existen diferentes medidas que podemos usar a la hora de especificar un tamaño en CSS.
Los tamaños los usamos para propiedades como **width**, **height**, **font-size** …
*Siempre que una medida se pueda especificar en píxeles, también se podrá en porcentaje y em y viceversa.

Porcentaje

Podemos especificar un porcentaje en un tamaño.

El elemento adoptará el porcentaje especificado de espacio de su contenedor o componente padre:

componente{width:100%} /*El componente ocupará todo el ancho de su contenedor*/

Píxeles

Especificamos los píxeles que ocupará (varían dependiendo de la resolución del monitor del usuario).

Especificar en formato: **npx** , donde **n** son los píxeles especificados:

componente{width:500px}

em

Especificamos cuanto ha de agrandarse o contraerse un componente, siendo **em** el tamaño actual y debemos indicarle delante un multiplicador en un formato **nem** , donde n es el multiplicador:

*Si **n** es < 1 , no indicar el 0 , directamente el punto:

componente{width:2em; height:.5em}

/*El ancho del componente será el doble y la altura será la mitad*/