



FULL STACK

Comenzamos en unos
minutos

Grids



CSS Grid

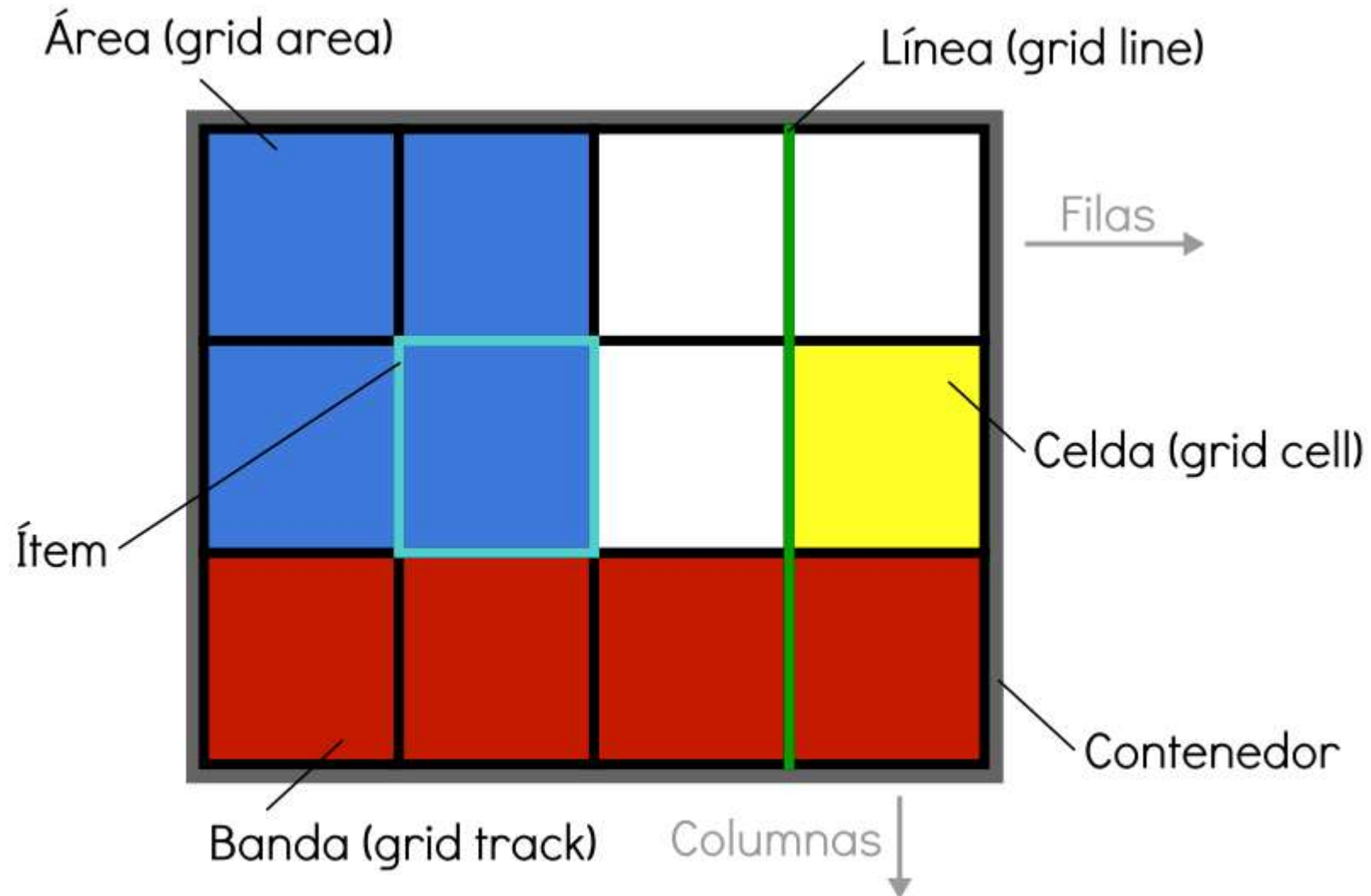
Grids

CSS Grid es un sistema de maquetación web que divide la página en una cuadrícula o rejilla (grid) a partir de la cual se pueden posicionar los diferentes elementos de manera más sencilla, versátil y coherente.

Su practicidad y sus múltiples ventajas lo han convertido en un estándar. Es decir, casi cualquier navegador soporta e interpreta este tipo de código.

La irrupción de CSS Grid, junto con Flexbox, supuso una revolución en el mundo de la programación web, ya que permitía realizar con mucho menos código elementos y estructuras que resultaban muy complejas o directamente imposibles.

Grids



Grids

Contenedor: El elemento padre contenedor que definirá la cuadrícula o rejilla.

Ítem: Cada uno de los hijos que contiene la cuadrícula (elemento contenedor).

Celda (grid cell): Cada uno de los cuadritos (unidad mínima) de la cuadrícula.

Area (grid area): Región o conjunto de celdas de la cuadrícula.

Banda (grid track): Banda horizontal o vertical de celdas de la cuadrícula.

Línea (grid line): Separador horizontal o vertical de las celdas de la cuadrícula.

Grids

Para utilizar cuadrículas Grid CSS, trabajaremos bajo el siguiente escenario:

```
<div class="grid"> <!-- contenedor -->
  <div class="a">Item 1</div> <!-- cada uno de los ítems del grid -->
  <div class="b">Item 2</div>
  <div class="c">Item 3</div>
  <div class="d">Item 4</div>
</div>
```

Grids

Ventajas de CSS Grid:

Mucha más flexibilidad: te permite controlar los elementos en las 2 dimensiones y con total libertad.

Menos código y menos bugs: se reduce considerablemente el código empleado, lo que conlleva revisiones más sencillas y menor probabilidad de bugs.

Optimización de recursos: al ser código más simple y consistente se economiza tanto en tiempo como en recursos necesarios para mostrar la página.

Responsive más sencillo: permite crear elementos dinámicos que se adaptan a diferentes tamaños o resoluciones sin complicaciones.

Grids

Como aplicar CSS Grid:

Para activar la cuadrícula grid hay que utilizar sobre el elemento contenedor la propiedad display y especificar el valor grid o inline-grid.

Tipo de elemento	Descripción
inline-grid	Establece una cuadrícula con ítems en línea, de forma equivalente a inline-block.
grid	Establece una cuadrícula con ítems en bloque, de forma equivalente a block.

Grids

Como aplicar CSS Grid:

Para activar la cuadrícula grid hay que utilizar sobre el elemento contenedor la propiedad display y especificar el valor grid o inline-grid.

Este valor influye en como se comportará la cuadrícula con el contenido exterior. El primero de ellos permite que la cuadrícula aparezca encima/debajo del contenido exterior (en bloque) y el segundo de ellos permite que la cuadrícula aparezca a la izquierda/derecha (en línea) del contenido exterior (ojo, la cuadrícula entera, no cada uno de sus ítems)

Tipo de elemento	Descripción
inline-grid	Establece una cuadrícula con ítems en línea, de forma equivalente a inline-block.
grid	Establece una cuadrícula con ítems en bloque, de forma equivalente a block.

Grids

Grid con filas y columnas:

Es posible crear cuadrículas con un tamaño explícito. Para ello, sólo tenemos que usar las propiedades CSS `grid-template-columns` y `grid-template-rows`, que sirven para indicar las dimensiones de cada celda de la cuadrícula, diferenciando entre columnas y filas. Las propiedades son las siguientes:

Propiedad	Valor	Descripción
<code>grid-template-columns</code>	<code>[<i>col1</i>] [<i>col2</i>] ...</code>	Establece el SIZE de cada columna (<i>col 1</i> , <i>col 2</i> ...).
<code>grid-template-rows</code>	<code>[<i>fila1</i>] [<i>fila2</i>] ...</code>	Establece el SIZE de cada fila (<i>fila 1</i> , <i>fila 2</i> ...).

Grids

Grid con filas y columnas:

```
.grid {  
  display: grid;  
  grid-template-columns: 50px 300px 1fr;  
  grid-template-rows: 200px 2fr 5% 1fr;  
}
```

Con estas propiedades definimos el numero de columnas(grid-template-columns) y filas(grid-template-rows). Cada valor que indiquemos será una columna o una fila. En el ejemplo tenemos 3 columnas y 4 filas. La unidad fraccionaria (fr) es una unidad relativa que nos permite definir un cierto porcentaje de la totalidad del contenedor dependiendo de los valores que ingresemos ejemplo, si tengo dos columnas (1fr, 3fr) se debe sumar las dimensiones que será la totalidad del contenedor y la primera columna ocupara solo la cuarta parte y la segunda columna ocupara tres cuartos de la totalidad del contenedor.

Espacios en Grids

Row-Gap:

Con esta propiedad podemos definir espacios entre filas solo debemos indicar un valor numerico, ejemplo `rows-gap:10px;`

Columns-Gap:

Con esta propiedad podemos definir espacios entre columnas solo debemos indicar un valor numerico, ejemplo `columns-gap:10px;`

Gap: Con esta propiedad podemos definir espacios entre filas y columnas solo debemos indicar un valor numero ya sea en unidades relativas (% ,em, rem,etc) y absolutas(px,etc). Esta propiedad nos permite definir un valor para los espacios de filas y otro para columnas.
ejemplo `gap:10px 20px;`

Grid-template-areas

Grid-template-areas:

Define una plantilla de cuadrícula haciendo referencia a los nombres de las áreas de cuadrícula que se especifican con la propiedad grid-area. La repetición del nombre de un área de la cuadrícula hace que el contenido abarque esas celdas. Un punto significa una celda vacía. La sintaxis en sí misma proporciona una visualización de la estructura de la cuadrícula.

Grid-area

Grid-area:

Esta propiedad Indica el nombre del área que se vinculara con el nombre que le indique en el grid-template-areas. Este atributo se aplica sobre ítems hijos del grid.

Grid-row

Grid-row-start:

Con esta propiedad podemos definir donde comienza un elemento con relación a las filas, solo debemos aplicarlo al mismo elemento y no al contenedor padre. Para eso utilizaremos las líneas guía definiendo el valor numérico

Grid-row-end:

Con esta propiedad podemos definir donde termina un elemento solo debemos aplicarlo al mismo elemento y no al contenedor padre. Para eso utilizaremos las líneas guía definiendo el valor numérico

Grid-row:

Con esta propiedad podemos definir el espacio que ocupa determinado elemento, se puede aplicar un valor numérico indicando el espacio que ocupara con relación a la fila. Ej: `grid-row: 3` ocupara tres filas. También podríamos indicar el valor de donde comienza y donde termina, ej: `grid-row: 2/4`; de esta manera comenzara en la 2 línea y terminara en la 4 línea, ocupando el espacio de 2 filas.

Grid-column

Grid-column-start:

Con esta propiedad podemos definir donde comienza un elemento en relación a las columnas, solo debemos aplicarlo al mismo elemento y no al contenedor padre. Para eso utilizaremos las líneas guía definiendo el valor numérico

Grid-column-end:

Podemos definir donde termina un elemento con relación a las columnas, solo debemos aplicarlo al mismo elemento y no al contenedor padre. Para eso utilizaremos las líneas guía definiendo el valor numérico

Grid-column:

Podemos definir el espacio que ocupa determinado elemento, se puede aplicar un valor numérico indicando el espacio que ocupara con relación a las columnas. Ej: `grid-column: 2;` ocupara dos columnas. También podríamos indicar el valor de donde comienza y donde termina, ej: `grid-row: 2/4;` de esta manera comenzara en la 2 línea y terminara en la 4 línea, ocupando el espacio de 2 columnas.

Alineaciones

Justify-items:

Distribuye los elementos en el eje horizontal. Estas propiedades se aplican sobre el elemento contenedor padre, pero afectan a los ítems hijos, por lo que actúan sobre la distribución de cada uno de los hijos. Los valores son: start | end | center | stretch

align-items:

Distribuye los elementos en el eje vertical. Estas propiedades se aplican sobre el elemento contenedor padre, pero afectan a los ítems hijos, por lo que actúan sobre la distribución de cada uno de los hijos. Los valores: start | end | center | stretch

Alineaciones

Justify- content:

Esta propiedad distribuye espacios y alinea de forma horizontal al conjunto de los elementos. Estas propiedades se aplican sobre el elemento contenedor padre, pero afectan a los ítems hijos, por lo que actúan sobre la distribución de cada uno de los hijos. Los valores son: start | end | center | stretch | space-around | space-between | space-evenly

align- content:

Esta propiedad distribuye espacios y alinea de forma horizontal al conjunto de los elementos. Estas propiedades se aplican sobre el elemento contenedor padre, pero afectan a los ítems hijos, por lo que actúan sobre la distribución de cada uno de los hijos. Los valores son: start | end | center | stretch | space-around | space-between | space-evenly

Alineaciones

Justify- self:

Con esta propiedad modifica la alineación del ítem hijo en el eje horizontal. Solo debemos aplicarlo sobre ese mismo elemento hijo y no al contenedor. Los valores que pueden tomar son los siguientes:
stretch | center | flex-start | flex-end | baseline

align- self:

Con esta propiedad modifica la alineación del ítem hijo en el eje horizontal. Solo debemos aplicarlo sobre ese mismo elemento hijo y no al contenedor. Los valores que pueden tomar son los siguientes:
stretch | center | flex-start | flex-end | baseline

Mobile First y Web Responsive



Responsive Web Design

Mobile First Web Design



Mobile First y Web Responsive

Que es mobile first?

se refiere a un modo de diseñar que tenga en cuenta, en primera instancia, un dispositivo móvil. Pantallas reducidas en comparación a los monitores que usamos normalmente con los ordenadores, y tras tener la maqueta preparada, realizar un escalado, es decir, aumentar el tamaño y adaptarlo a una pantalla de escritorio.

Web responsive

El diseño responsive, es la filosofía de diseño opuesta, es un estándar. Este tipo de páginas web, son adaptativas, es decir, al reducir la resolución se reduce el tamaño del contenido, y es, realmente, lo que tiene Google bajo su lupa. Que nuestro sitio web sea responsive no es opcional sino obligatorio, de lo contrario nuestra visibilidad y efectividad SEO se verá reducida ya que estaremos perdiendo todas las visitas de potenciales clientes que usen dispositivos móviles para navegar.

Media queries



Media Queries

Media queries

Que son las media queries?

Las media queries (en español "consultas de medios") son útiles cuando deseas modificar tu página web o aplicación en función del tipo de dispositivo (como una impresora o una pantalla) o de características y parámetros específicos (como la resolución de la pantalla o el ancho del viewport del navegador).

Ejemplo:

```
@media screen and (min-width: 500px) and (max-width: 800px) { ... }
```

Metaviewport



Meta Viewport

```
<meta name="viewport"  
content="width=device-  
width, initial-  
scale=1.0">
```



Metaviewport

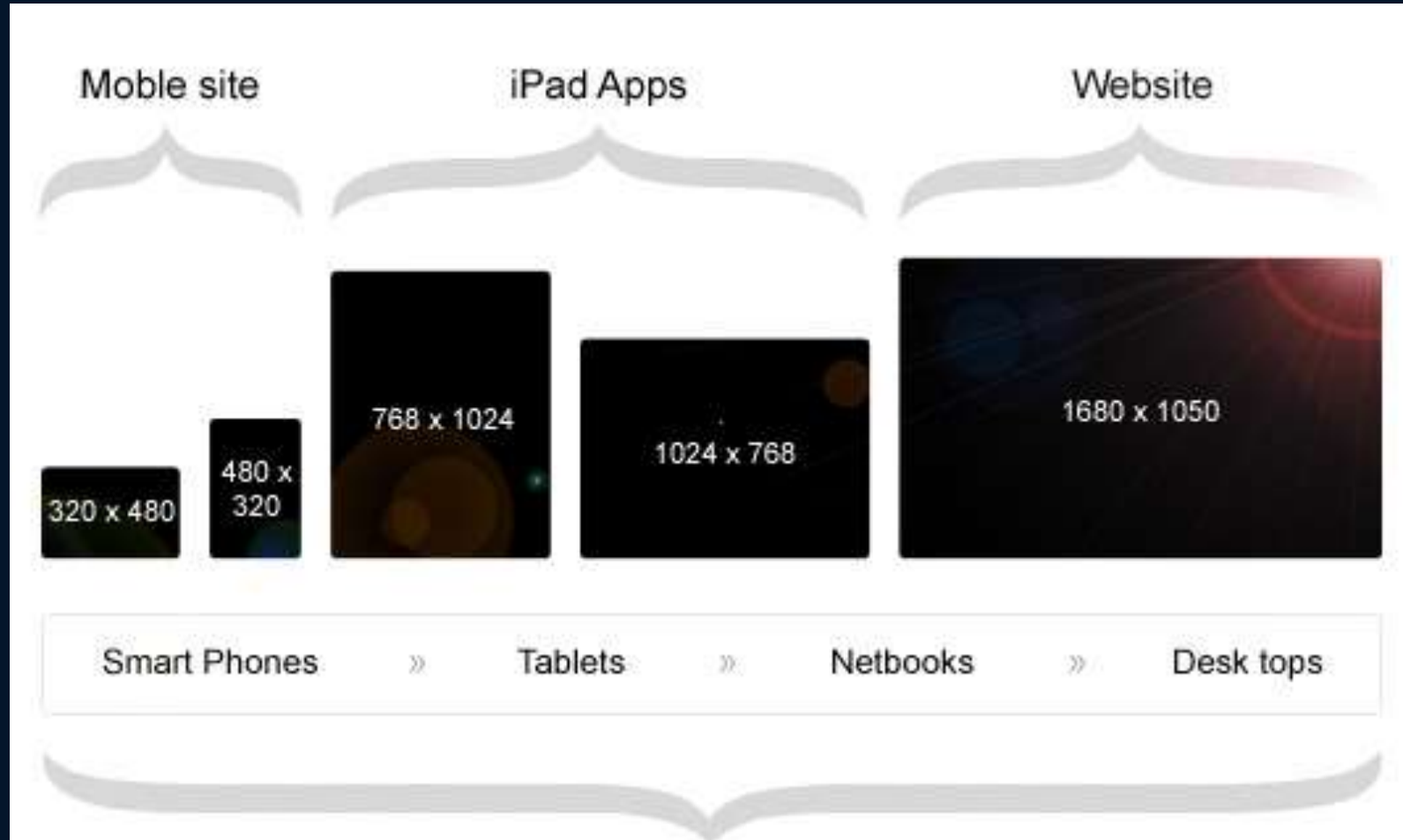
Que es Metaviewport?

se refiere a la etiqueta que mejor representa la web en movilidad, ya que nos permite indicar cómo se verá un proyecto web en los dispositivos móviles. Apple la creó para decirle al iPhone como debería renderizar el documento, convirtiéndose desde entonces en un estándar que ya está soportado por la mayoría de dispositivos.

Como definición rápida, diremos que viewport podría traducirse como vista o ventana y nos sirve para definir qué área de pantalla está disponible al renderizar un documento, nivel de escalado y el zoom que debe mostrar inicialmente. Todo ello, con parámetros que le damos a la propia etiqueta META.

```
<meta name="viewport" content="user-scalable=no,  
width=device-width, initial-scale=1">
```

Break points



Break points

Que son los Break points?

Son las medidas de anchura en donde se realizan saltos para el diseño responsive y se aplican los estilos CSS concretos para unas determinadas media queries. Es decir, los breakpoints son los saltos en los que la pantalla cambia de layout.

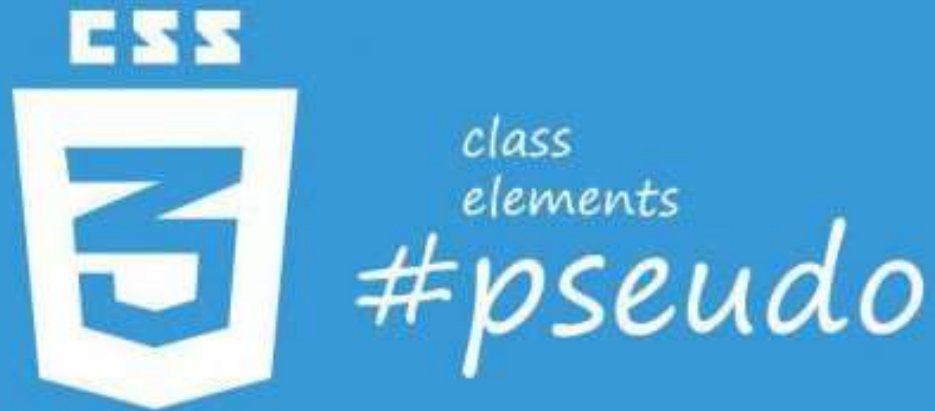
Existen unos breakpoints estandarizados que debemos de tener en cuenta:

Móviles: entre 320 y 480 píxeles.

Tablets: entre 768 y 1024 píxeles.

Pantallas grandes: más de 1200 píxeles.

Pseudoclasses para enlaces



Pseudoclases para enlaces

a:active

se refiere a aquellos enlaces en las que se han hecho clic y están activos.

a:link

sirve para apuntar a enlaces normales de la página web (no accedidos).

a:hover

Definimos el estilo para enlaces cuando el cursor se encuentra sobre dichos enlaces

a:visited

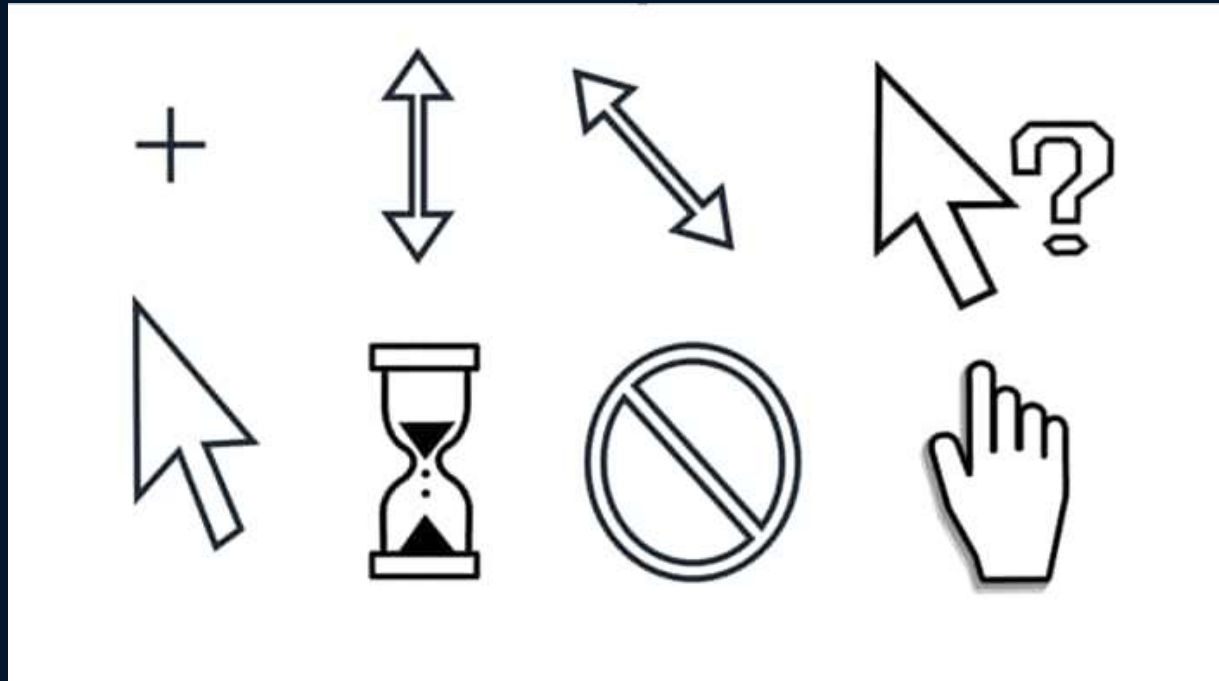
apuntaremos a los enlaces visitados o accedidos por el usuario.

😊 Pseudoclase

😊 Selector

La propiedad Cursor

La propiedad cursor nos permite establecer la forma de cursor que queremos sobre nuestro elemento y puede tomar una rica variedad de valores que definen una forma en específico.



La propiedad Cursor

default:

es el puntero por defecto, con forma de flecha



help:

cursor de ayuda, con forma de flecha acompañado de un signo de interrogación.



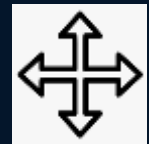
crosshair:

puntero en forma de cruz.



move:

cursor en forma de cruz pero con flechas en cada extremo.



pointer:

























en forma de manito con el índice señalando.



😊 Valores
😊 Definición

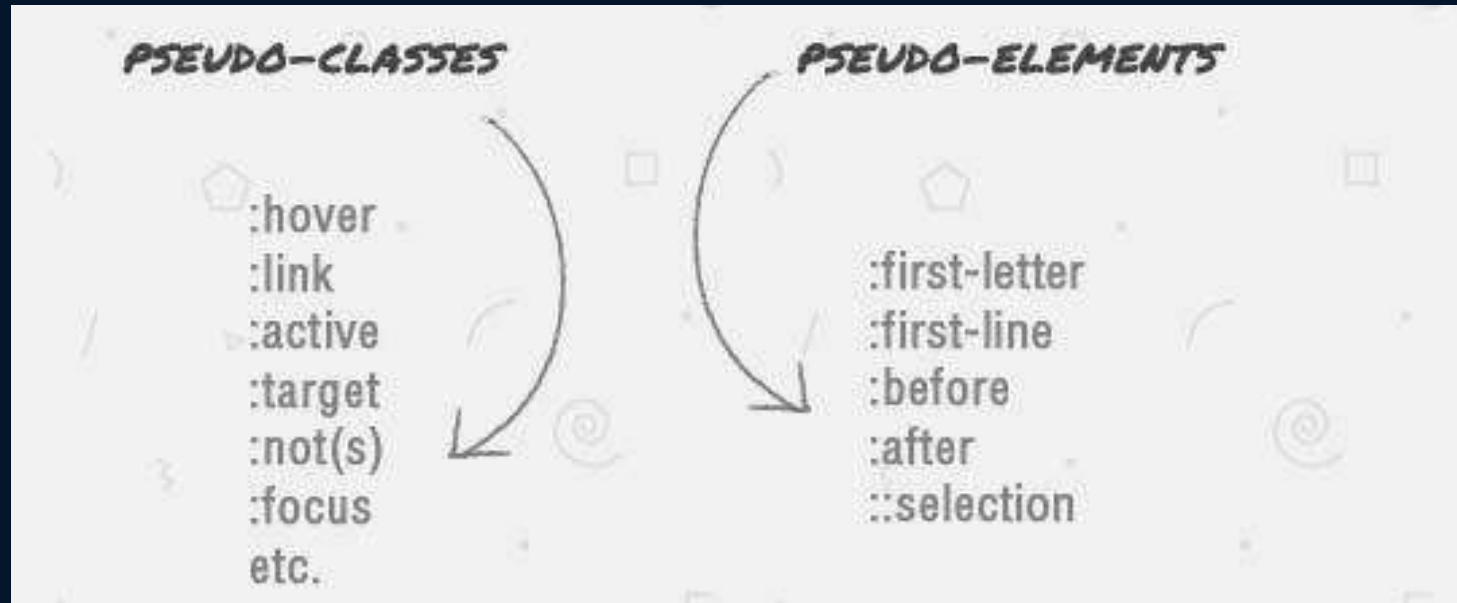
La propiedad Cursor

Más valores para la propiedad cursor

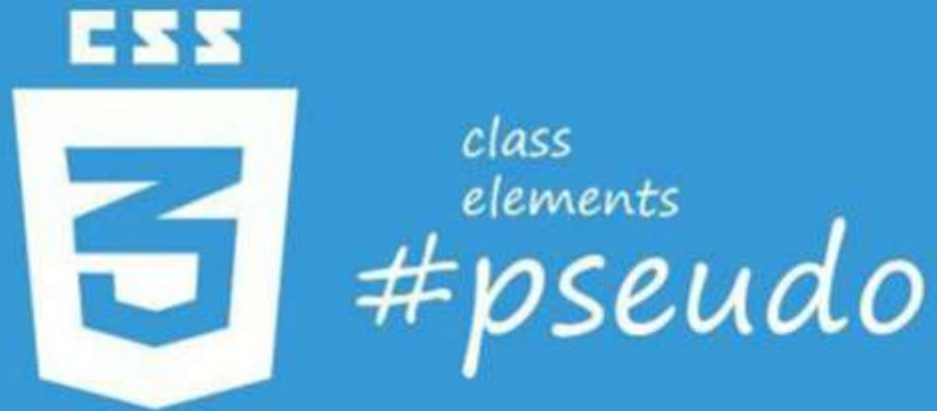
 auto	 move	 no-drop	 col-resize
 all-scroll	 pointer	 not-allowed	 row-resize
 crosshair	 progress	 e-resize	 ne-resize
 default	 text	 n-resize	 nw-resize
 help	 vertical-text	 s-resize	 se-resize
 inherit	 wait	 w-resize	 sw-resize

PseudoClases y PseudoElementos

Comúnmente se suele decir indistintamente pseudo-clases o pseudo-elementos, pero aunque su objetivo sea el mismo, el de ampliar nuestra capacidad de especificar aquello que necesitamos seleccionar cuando no podemos hacerlo a través de los selectores habituales (class, ID o selector de atributos), hay una leve diferencia entre ambos. Mientras los pseudo-elementos aplican para seleccionar o agregar contenido a un elemento, las pseudo-clases nos permiten seleccionar dicho elemento bajo determinadas condiciones.



PseudoClases y PseudoElementos



PseudoClases y PseudoElementos

Las pseudoclasses: son selectores que básicamente nos permiten apuntar a un elemento específico que cumple una característica determinada. Una pseudoclase está compuesta por una palabra clave que se escribe empezando con dos puntos simples; por ejemplo: `:first-child` o `:last-child`

Los pseudo-elementos: están pensados para establecer un estilo a partes específicas de un elemento, por ejemplo: la primera línea de un texto, la primera letra de un texto, etc. veamos cuáles son y la forma de aplicarlos. Un pseudoelemento está formado por una palabra clave antecedita por dos puntos dobles; por ejemplo: `::first-line`, a continuación conozcamos los agregados en CSS3.

PseudoElementos

::after	Inserta algo después del contenido del elemento
::before	Inserta algo antes del contenido del elemento
::first-letter	Selecciona la primer del elemento
::first-line	Selecciona la primer linea del elemento
::selection	Selecciona la parte de un elemento seleccionada por el usuario

PseudoClases

:active	Selecciona el link activo
:checked	Selecciona los <input> de tipo radio o checkbox como también <option> dentro de un select que está marcado o conmutado a un estado <i>on</i>
:empty	Selecciona cada elemento seleccionado que no tiene hijos
:first-child	Selecciona cada elemento seleccionado que sea primer hijo de su padre. Ver ampliado
:first-of-type	Selecciona el primer elemento de tipo seleccionado dentro de un contenedor. Ver ampliado
:focus	Selecciona el elemento cuando esta en foco
:hover	Selecciona enlaces al pasar el ratón por encima

PseudoClases

:invalid	Selecciona todos los elementos seleccionados con un valor no válido
:last-child	Selecciona el ultimo elemento seleccionado que sea primer hijo de su padre. Ver ampliado
:last-of-type	Selecciona el ultimo elemento de tipo seleccionado dentro de un contenedor. Ver ampliado
:link	Selecciona todos los enlaces no visitados
:not(selector)	Selecciona cualquier elemento que no sea el seleccionado
:nth-child(n)	Selecciona uno o más elementos en función de su posición entre un grupo de hermanos
:optional	Selecciona elementos sin el atributo "required"

PseudoClases y PseudoElementos

Las pseudoclasses:

- : :first-child
- ::last-child
- ::checked
- ::default
- ::empty
- ::focus
- ::hover
- ::nth-child
- ::nth-last-child
- ::nth-last-of-type
- ::nth-of-type
- ::placeholder
- ::required

Los pseudo-elementos:

- ::after
- ::before
- ::first-letter
- ::first-line
- ::selection

Transiciones en CSS



Transiciones en CSS

Una transición es el evento que sucede entre el estado inicial y final, es decir el cambio de estado, dicho cambio puede ser de alguna característica o propiedad de un elemento HTML en nuestro caso, este cambio lo haremos mediante código CSS.



Transiciones en CSS

Cómo poner transiciones en CSS3

Poner una transición a cualquier elemento es muy simple, mediante el selector apuntaremos al elemento que queremos aplicar la transición, y mediante las propiedades de transición determinamos cuál es la propiedad a cambiar, la duración de dicha transición, su función de tiempo, retraso, etc.

La sintaxis tendría la siguiente forma: `selector {propiedades: valores;}`

Cómo funcionan las transiciones

Las transiciones en CSS3 funcionan de manera bastante simple, mediante un selector estableceremos el elemento que será afectado, es decir al que queremos aplicar una transición, luego mediante las propiedades de transición estableceremos que características y como se realizará la transición.

Transiciones en CSS

Propiedades de Transiciones:

Transition-property:

Se usa para definir los nombres de las propiedades CSS en las que el efecto de la transición debe aplicarse o aplicar el valor all que se aplique para todas.

Transition-duration:

Establece el tiempo que debe tardar una animación de transición en completarse. Por defecto, el valor es de 0s, esto quiere decir que no se producirá ninguna animación.

Transition-timing-function:

Esta propiedad establece cómo se calculan los valores intermedios para las propiedades CSS afectadas por un efecto de transición. Los posibles valores son ease | ease-in | ease-out | ease-in-out | cubic-bezier

Transition-delay:

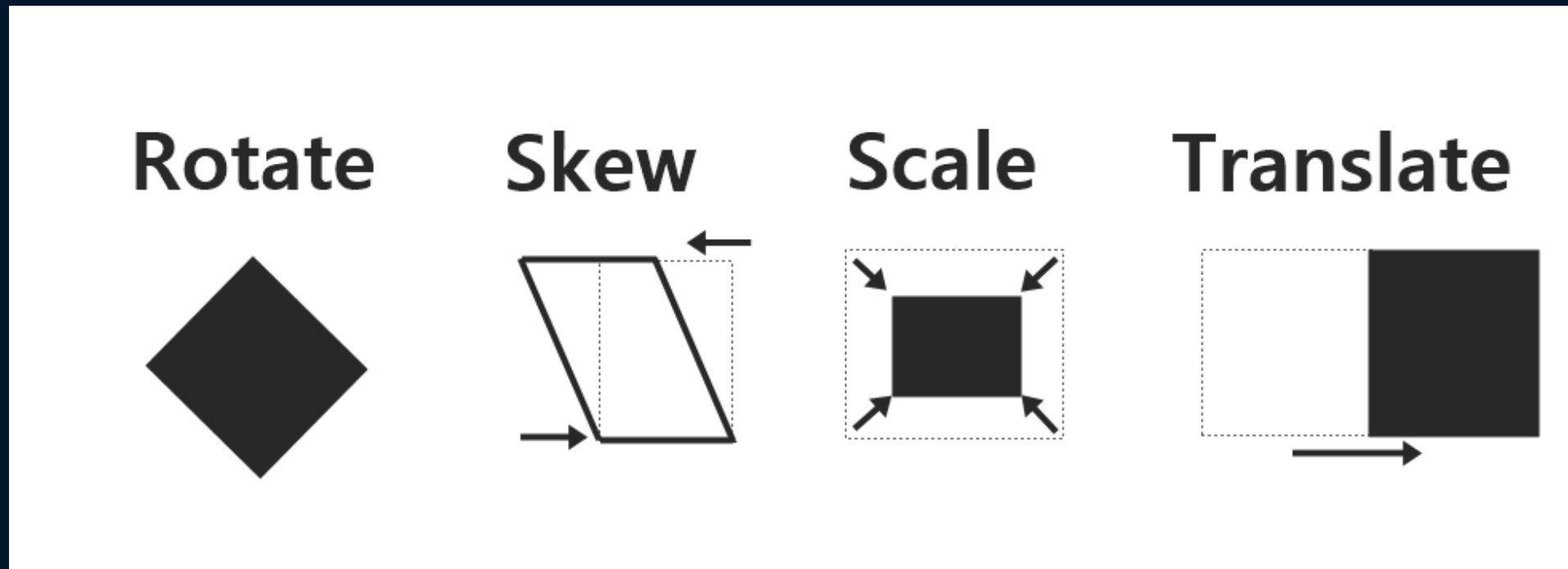
Especifica la cantidad de tiempo a esperar entre un cambio pedido hacia una propiedad y el comienzo de un efecto de transición (transition effect). Los valores pueden ser de 0s o 0ms en adelante

Transformaciones en CSS



Transformaciones en CSS

Las transformaciones de CSS3 son cambios de una determinada característica de un elemento que se encuentra dentro de la estructura del documento HTML, dichos cambios se realizan de forma controlada mediante las propiedades de transformación CSS3. Entre las transformaciones más conocidas podemos mencionar: rotación, translación, escala.



Transformaciones en CSS

scale(): Afecta el tamaño del elemento. Esto también se aplica a la font-size, padding, height, y width de un elemento, también. También es una función abreviada para las funciones scaleX y scaleY.

skewX() y skewY(): inclina un elemento hacia la izquierda o hacia la derecha, como convertir un rectángulo en un paralelogramo. skew() es una taquigrafía que combina skewX() y skewY acepta ambos valores.

translate(): Mueve un elemento hacia los lados o hacia arriba y hacia abajo.

rotate(): Gira el elemento en el sentido de las agujas del reloj desde su posición actual.

matrix(): Una función que probablemente no esté destinada a escribirse a mano, pero que combina todas las transformaciones en una.

perspective(): No afecta al elemento en sí, pero afecta las transformaciones de las transformaciones 3D de los elementos descendientes, lo que les permite tener una perspectiva de profundidad consistente.

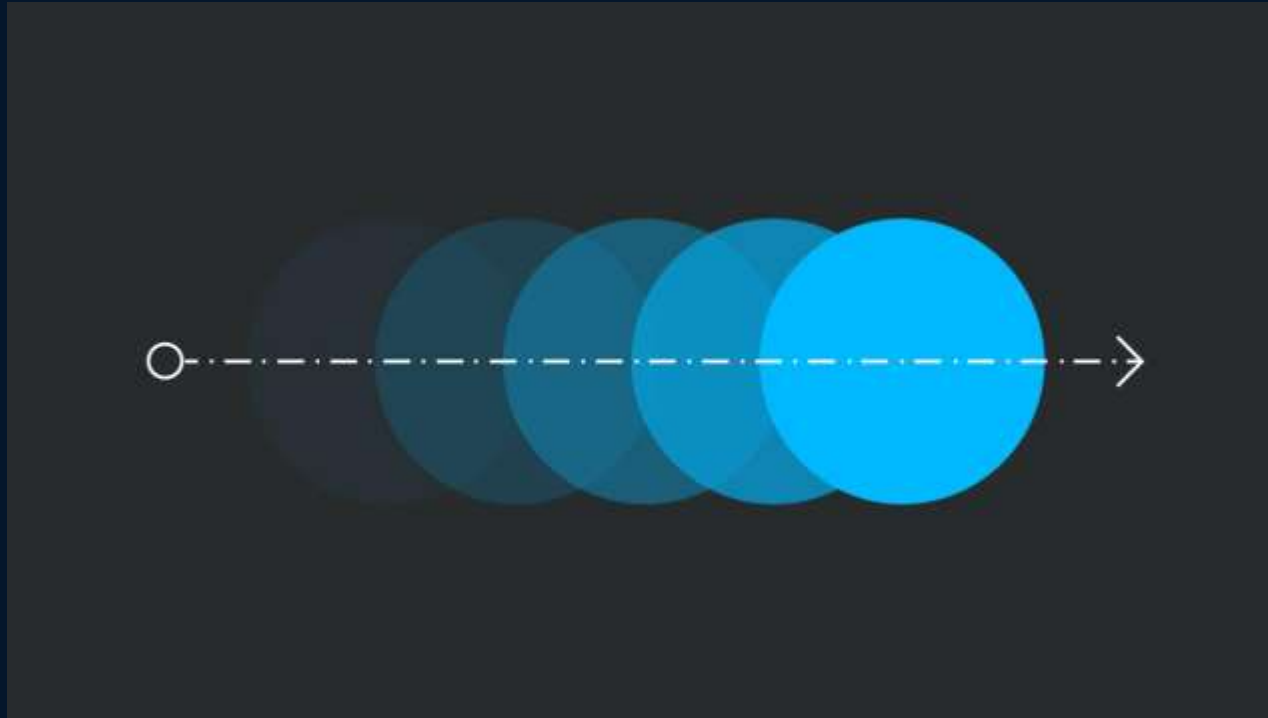
Animaciones en CSS

CSS Animation



Animaciones en CSS

Las animaciones son una nueva característica introducida en CSS3, gracias a ellas ahora podemos aplicar animaciones a cualquier elemento, básicamente consiste en cambiar el estilo de un elemento de forma controlada; por ejemplo podemos hacer que un elemento cambie de color, dimensión, o cualquier propiedad que admita de acuerdo al tipo de elemento que sea.



Animaciones en CSS

animation-name: Con esta propiedad definimos el nombre de la animación

animation-duration: Con esta propiedad indicamos la duración de la animación en segundos o ms. Ejemplo 0.5s.

animation-timing-function: Con esta esta propiedad indicamos el tiempo a lo largo de esa animación por medio de los siguientes valores ease | linear | ease-in | ease-out | ease-in-out | cubic-bezier(A, B, C, D).

animation-delay: Definimos el retraso de esa animación. Ej 1s.

Animaciones en CSS

animation-iteration-count: Indicamos el numero de veces a repetirse la animacion y los valores serian por ejemplo: 1 | 2 | infinite

animation-direction: Indicamos la dirección de la animación normal | reverse | alternate | alternate-reverse

animation-fill-mode: Podemos indicar que debe mostrar la animación cuando ha finalizado y ya no se está reproduciendo; si mostrar el estado inicial (backwards), el estado final (forwards) o una combinación de ambas (both).

animation-play-state: Nos permite establecer la animación a estado de reproducción (running) o pausarla (paused).

Animaciones en CSS

animation-iteration-count: Indicamos el numero de veces a repetirse la animacion y los valores serian por ejemplo: 1 | 2 | infinite

animation-direction: Indicamos la dirección de la animación normal | reverse | alternate | alternate-reverse

animation-fill-mode: Podemos indicar que debe mostrar la animación cuando ha finalizado y ya no se está reproduciendo; si mostrar el estado inicial (backwards), el estado final (forwards) o una combinación de ambas (both).

animation-play-state: Nos permite establecer la animación a estado de reproducción (running) o pausarla (paused).

Fotogramas (keyframes)

Ya sabemos como indicar a ciertas etiquetas HTML que reproduzcan una animación, con ciertas propiedades. Sin embargo, nos falta la parte más importante: definir los fotogramas de dicha animación. Para ello utilizaremos la regla `@keyframes`, la cuál es muy sencilla de utilizar y se basa en el siguiente esquema:

```
@keyframes nombre {  
  selectorkeyframe {  
    propiedad : valor ;  
    propiedad : valor  
  }  
}
```

Fotogramas (keyframes)

Ya Definido el nombre hacemos uso de la regla @keyframes aplicando ese nombre separado por un espacio después abrimos las llaves y dentro incorporamos cada uno de los puntos de la línea de la animación por medio de los porcentajes comprendidos entre 0% y 100%. Dentro de esos porcentajes abrimos llaves nuevamente y aplicamos todas las propiedades y valores correspondientes para ese punto.

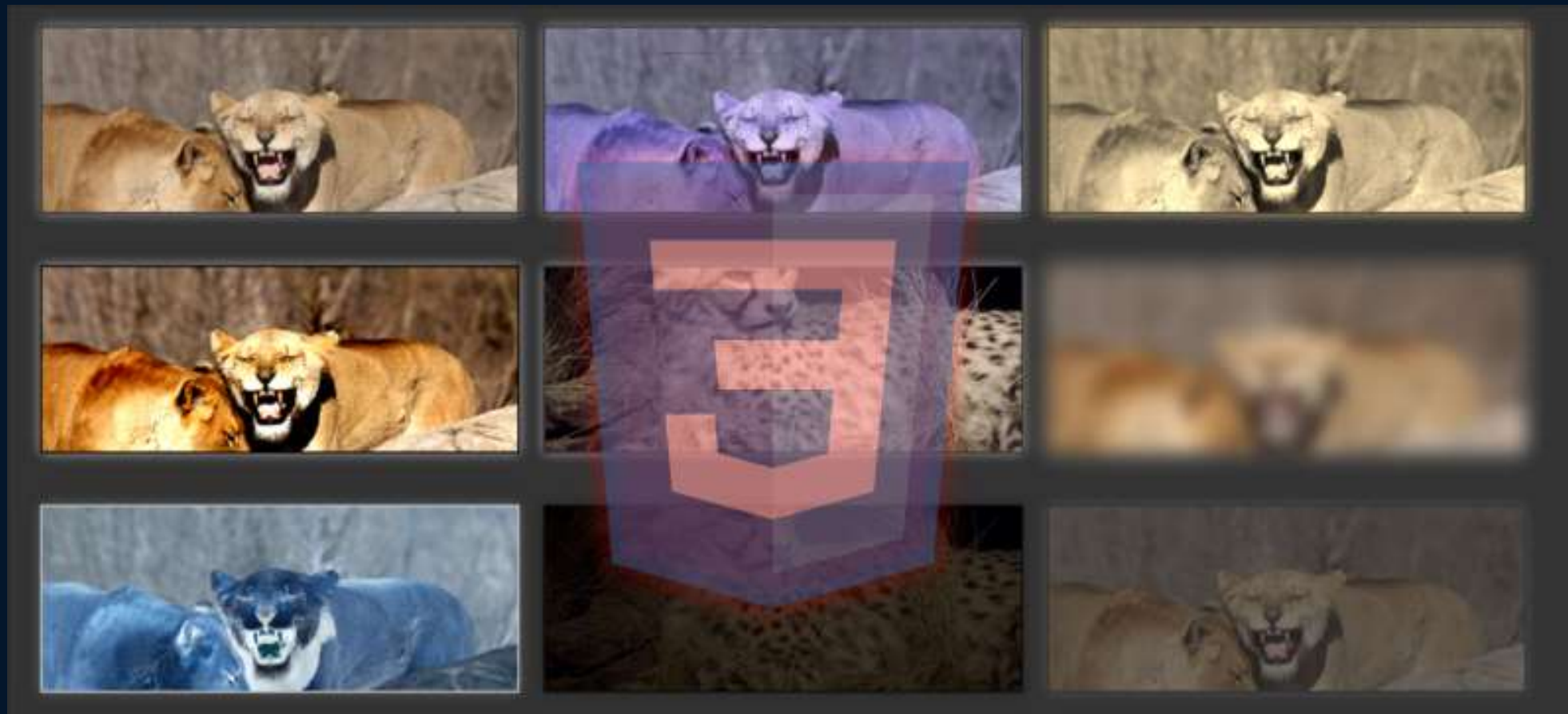
```
body{
  padding: 50px;
}

.box{
  background-color: red;
  width: 200px;
  height: 200px;
  position: relative;
  display: inline-block;

  animation-name: myframes;
  animation-duration: 2s;
  animation-iteration-count: infinite;
  animation-timing-function: ease-in-out;
}

@keyframes myframes{
  0%   {top: 0px; left: 0px; background: pink;}
  25%  {top: 0px; left: 100px; background: purple;}
  50%  {top: 100px; left: 100px; background: black;}
  75%  {top: 100px; left: 0px; background: tomato;}
  100% {top: 0px; left: 0px; background: gold;}
}
```

Filtros en CSS



Filtros en CSS

Qué son y para qué sirven los filtros de CSS3

Los filtros en CSS3 son funcionalidades que nos permiten aplicar efectos visuales a un elemento, en nuestro caso específico a una imagen. Por ejemplo, mediante un código de CSS podemos aplicar una escala de grises a una imagen para que esta aparezca en blanco y negro, esta misma imagen podemos utilizar en otra parte de la página con un filtro de sepia, entonces la misma imagen aparecerá en colores marrón- rojizo.

¿Como aplicar un filtro?

Colocar un filtro es muy sencillo, simplemente apuntamos al elemento y definimos la propiedad filter, en este caso también estableceremos una función de filtro, existen varios, ya están establecidos y solo tenemos que aprenderlas, por ejemplo la función de filtro para escala de grises es: grayscale;

Filtros en CSS

Tipos de filtros de CSS3

```
#imagen{  
  filter: grayscale(50%);  
}
```

```
#imagen{  
  filter: sepia(0.5);  
}
```

```
#imagen{  
  filter: blur(4px);  
}
```

Grayscale()
Sepia()
Blur()
Brightness()
Drop-shadow()
Hue-rotate()
Invert()
None()
Opacity()
Saturate()

```
#imagen{  
  filter: hue-rotate(120deg);  
}
```

```
#imagen{  
  filter: brightness(70%);  
}
```

```
#imagen{  
  filter: opacity(0.5);  
}
```


Filtros en CSS

Tipos de filtros de CSS3

Grayscale (aplica escala de grises)
Sepia (aplica sepia)
Blur(aplica desenfoque)
Brightness(modifica el brillo)
Drop-shadow(aplica sombra con desenfoque)
Hue-rotate(rota la matriz de colores)
Invert(invierte los colores)
None(no aplica ningún filtro)
Opacity(aplica transparencia)
Saturate(aplica saturación)

Sitios recomendados



Donde descargar iconos

<https://fontawesome.com/>

<https://www.flaticon.com/>

<https://www.fontisto.com/>

<https://ikonate.com/>

Sitios de colores

<https://material.io/>

<https://www.design-seeds.com/>

<https://www.toptal.com/designers/colourcode/>

<https://colorsupplyyy.com/app>

<https://encycolorpedia.es/>

<https://color.adobe.com/es/create/color-wheel>

<http://khroma.co/>

<http://www.patternify.com/>

Sitios de ilustraciones

<https://pixabay.com/es/illustrations/>

<https://www.pexels.com/es-es/>

<https://www.freepik.com/>

<https://www.humaaans.com/>

<https://undraw.co/illustrations>

<https://absurd.design/>

<https://drawkit.com/>

<https://pixabay.com/es/vectors/search/>

<https://icons8.com/illustrations>

<http://www.patternify.com/>

Optimizador de imágenes:

<https://www.resizepixel.com/>

Sitios de Tipografías

<https://www.designermill.com/category/free-fonts/>

<https://www.myfonts.com/>

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

<https://www.fontpair.co/>

<https://lingojam.com/CoolTextFonts>

<https://fontjoy.com/>

<https://typ.io/>

Sitios diseño creatividad e inspiración

<https://www.designspiration.com/>

<https://www.awwwards.com/>

<https://dribbble.com/>

<https://www.behance.net/galleries/ui-ux/ui-ux>

<https://collectui.com/>

<https://mobbin.design/browse/ios/apps>

<https://muz.li/>

Sitios para crear favicon

<https://www.favicon.cc/>

<https://realfavicongenerator.net/>

<https://genfavicon.com/es/>

Sitios referentes al SEO

https://developers.google.com/search/docs/beginner/seo-starter-guide?hl=en&visit_id=637689044845753795-546123681&rd=1

<https://moz.com/beginners-guide-to-seo>

<https://neilpatel.com/what-is-seo/>

<https://www.wordstream.com/meta-tags>

https://developers.google.com/search/docs/advanced/crawling/special-tags?hl=en&visit_id=637689046770838748-2744182862&rd=1

<https://yoast.com/meta-descriptions/>

<https://moz.com/learn/seo/on-page-factors>

Sitios referentes al SEO

<https://bloo.media/blog/mejores-herramientas-seo/>