



Programador Web Inicial Front- End Developer

CSS y maquetado avanzado

CSS (parte 3)

Presentación

En esta unidad vemos las propiedades de CSS más avanzadas que nos ayudarán en la maquetación del sitio.

Objetivos

Que los participantes logren...

- Conocer más propiedades de CSS.
- Entender y utilizar las distintas transformaciones.
- Conocer para qué sirven las media queries.

Bloques temáticos

1. Transformaciones.
2. Transiciones.
3. Media queries.

1. Transformaciones

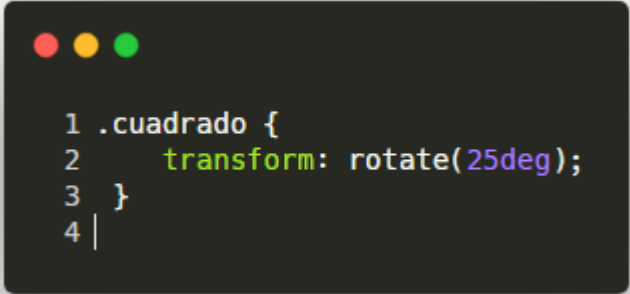
La propiedad CSS transform permite manipular el sistema de coordenadas de un elemento usando las funciones de transformación.

rotate

La función CSS rotate() define una transformación que gira un elemento alrededor de un punto fijo en un plano 2D sin deformarlo.

El punto fijo alrededor del cual gira el elemento, mencionado anteriormente, también se conoce como el origen de transformación. Se establece de manera predeterminada en el centro del elemento.

```
transform: rotate(deg);
```



```
1 .cuadrado {  
2   transform: rotate(25deg);  
3 }  
4 |
```

scale

La función CSS scale() define una transformación que modifica el tamaño de un elemento en el plano 2D. Debido a que la cantidad de escalado está definida por un vector, puede cambiar el tamaño de las dimensiones horizontal y vertical a diferentes escalas.

```
scale(sx) scale(sx, sy)
```

```
1 .circulo {  
2     transform: scale(2.5);  
3 }  
4  
5 .cuadrado {  
6     transform: scale(2.5,4); /*x,y*/  
7 }
```

translate

La función de CSS `translate()` recoloca un elemento en el eje horizontal y/o vertical.

`transform: translate(tx, ty);`

```
1 .circulo {  
2     transform: translate(50px, 100px); /*x,y*/  
3 }  
4  
5 .cuadrado {  
6     transform: translateX(100px); /*x*/  
7 }
```

2. Transiciones

Las transiciones CSS son pequeños cambios en propiedades de la hoja de estilos desencadenados por acontecimientos generados por interacciones del usuario, como por ejemplo cuando el mouse pasa por encima de algo (:hover) el cambio se ve tosco. En una transición, estos cambios en las propiedades se producen de manera progresiva durante un intervalo de tiempo.

Estructura

Transition: [property] [duration] [timing-function] [delay];

Transition: all 0.3s ease 0.5s;

transition-property: Especifica el nombre o nombres de las propiedades CSS a las que deberían aplicarse las transiciones. Sólo las propiedades que se enumeran aquí son animadas durante las transiciones; los cambios en el resto de las propiedades suceden de manera instantánea como siempre. En caso que sean varias propiedades se puede usar el valor all.

transition-duration: Especifica la duración en la que sucederán las transiciones.

transition-timing-function: Especifica la curva cúbica bézier que se usa para definir cómo se computan los valores intermedios para las propiedades.

transition-delay: Define el tiempo de espera entre el momento en que se cambia una propiedad y el inicio de la transición.

Algunas propiedades que pueden ser animadas: background-color, background-image, background-position, background-size, border-color, border-radius, color, font-size, opacity y entre otras.


```

1 .border {
2     width:500px;
3     height:300px;
4     background:#676470;
5     color:#fff;
6     transition:all 0.3s ease;
7 }
8
9 .border:hover{
10     box-shadow: inset 0 0 0 25px #53a7ea;
11 }

```

Propiedad transition-timing-function

Valor	Descripción
linear	La animación tiene la misma velocidad de principio a fin.
ease	La animación tiene un comienzo lento, luego rápido, antes de que termine lentamente.
ease-in	La animación tiene un comienzo lento.
ease-out	La animación tiene un final lento.
ease-in-out	La animación tiene un comienzo lento y un final lento.
step-start	Equivalente a pasos (1, inicio)
step-end	Equivalente a pasos (1, fin)

steps(int,start end)	Especifica una función paso a paso, con dos parámetros. El primer parámetro especifica el número de intervalos en la función. Debe ser un número entero positivo (mayor que 0). El segundo parámetro, que es opcional, es el valor "inicio" o "final", y especifica el punto en el que se produce el cambio de valores dentro del intervalo. Si se omite el segundo parámetro, se le asigna el valor "fin"
cubic-bezier(n,n,n,n)	Defina sus propios valores en la función cubic-bezier Los valores posibles son valores numéricos de 0 a 1

3. Media queries

Media queries significa "consulta de medios" y son útiles cuando deseamos modificar el aspecto de nuestra página basados en la resolución de la pantalla o el ancho del viewport del navegador.



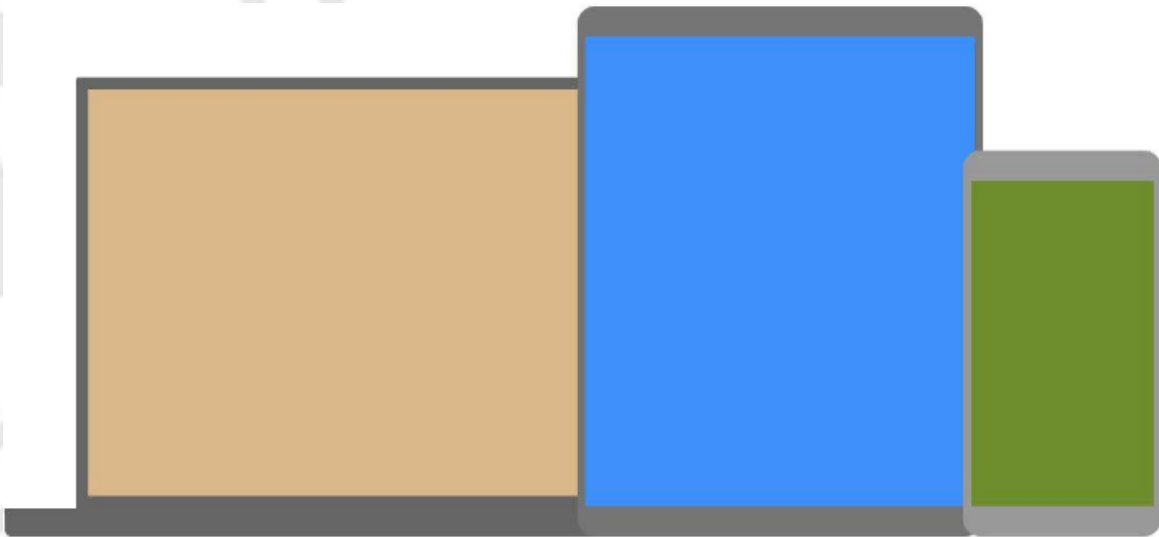
Imagen: <https://www.silocreativo.com/media-queries-css/>

El resultado de la consulta es "verdadero" cuando el tipo de medio (si se especifica) coincide con el dispositivo en el que se está mostrando el documento y todas las expresiones en la media query son "verdaderas". En este caso, se aplican los estilos correspondientes, siguiendo las reglas usuales de cascada.

```
1 @media (min-width: 700px){  
2   /*Especifico la lista de estilos que cambian;*/  
3 }
```



Imagen: <https://www.silocreativo.com/media-queries-css/>



```
/* Establecer el color de fondo del cuerpo en beige */
body {
  background-color: tan;
}

/* En pantallas de 992px o menos, establezca el color de fondo en
azul */
@media screen and (max-width: 992px) {
  body {
    background-color: blue;
  }
}

/* En pantallas de 600 px o menos, establezca el color de fondo en
verde */
@media screen and (max-width: 600px) {
  body {
    background-color: olive;
  }
}
```

Forma alternativa de definir nuestros media queries

Una forma no tan usada, pero muchas veces más beneficiosa para el usuario, de definir nuestros media queries es usando el atributo media de la etiqueta link que usamos para incluir nuestros estilos.

Mediante el uso de esta técnica y la correcta separación de las hojas de estilos podemos reducir drásticamente la cantidad de datos que el usuario necesite descargar para ver correctamente nuestro sitio.



```
<!-- este archivo solo será descargado por dispositivos con 800px o menos de  
ancho de pantalla -->  
<link rel="stylesheet" media="(max-width: 800px)" href="example.css" />
```

¿En qué se diferencian min-width y max-width?

Las directivas min-width y max-width se usan para acotar las resoluciones en donde se van a aplicar los estilos que definamos dentro de nuestra media query. Mientras min-width especifica la resolución mínima desde la cual se aplican los estilos, max-width define la resolución máxima dentro de la cual serán válidos los estilos.



```
/* Si el tamaño de la pantalla es de 601px o más, ajusta el tamaño de la fuente  
de <div> a 80px */  
@media only screen and (min-width: 601px) {  
  div.example {  
    font-size: 80px;  
  }  
}  
  
/* Si el tamaño de la pantalla es de 600px o menos, ajusta el tamaño de la  
fuente de <div> a 30px */  
@media only screen and (max-width: 600px) {  
  div.example {  
    font-size: 30px;  
  }  
}
```

¿Qué es Mobile First?

Es una forma de pensar nuestras hojas de estilo aplicando primero los estilos para pantallas más chicas, que por lo general tienen las conexiones de datos más lentas y se verán beneficiados por esta técnica.

Esto significa que debemos realizar algunos cambios en nuestro CSS.

En lugar de cambiar los estilos cuando el ancho es menor a 768px, deberíamos cambiar el diseño cuando el ancho es mayor a 768px. Esto hará que nuestro diseño sea Mobile First.

Bibliografía utilizada y sugerida

Libros y otros manuscritos:

Collell Puig, Jordi. CSS3-y-Javascript-Avanzado. Universidad Oberta de Catalunya.

Gauchat, Juan Diego. El gran libro de HTML5, CSS3 y Javascript. 1era Edición. 2012

Artículos de revista en formato electrónico:

Mozilla MDN Web Docs. Disponible desde la URL:

<https://developer.mozilla.org/es/>

w3schools. Disponible desde la URL:

https://www.w3schools.com/css/css3_mediaqueries_ex.asp