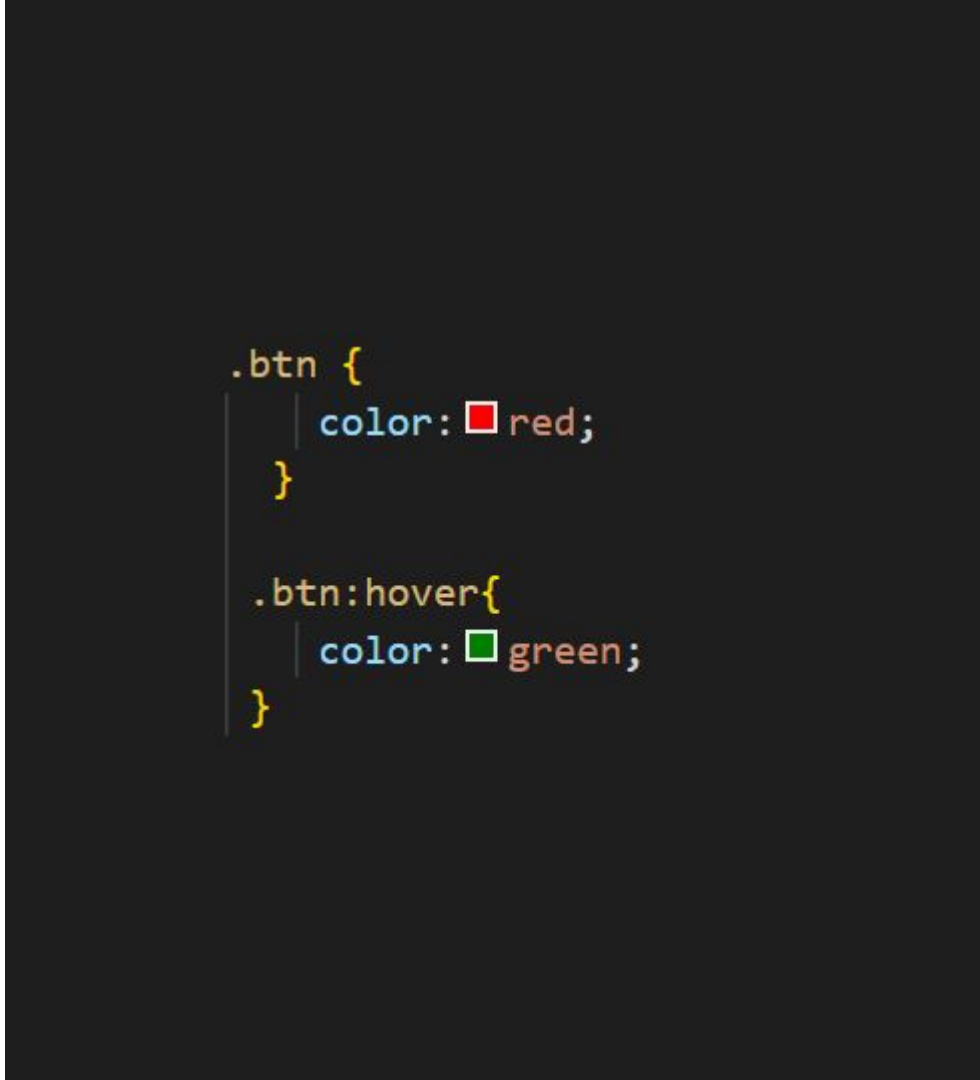
Pseudoclase Hover + efectos

Edición Web I

Pseudoclase Hover

Las **pseudoclases** se utilizan para hacer referencia a elementos HTML que tengan un cierto comportamiento o estado concreto.

.nombre-clase:pseudoclase



Pseudoclase de interacción (links)

:link
Estado por defecto, enlace no visitado.

:visited
Una vez el enlace es visitado, lo guarda en el historial del navegador.

:hover
Cuando pasamos el cursor por encima

:focus
Cuando el elemento tiene el foco o hacemos click

:active
Cuando está activo

```css
.btn {
    color: red;
 }


 .btn:visited{
    color: green;
}


 .btn:hover{
    color: rgb(235, 10, 138);
}

.btn:focus{
    color: rgb(15, 245, 192);
}


.btn:active{
    color: rgb(252, 96, 13);
}
```

Pseudoclase de interacción (links)

:link
Estado por defecto, enalce no visitado.

:visited
Una vez el enlace es visitado, lo guarda en el historial del navegador.

:hover
Cuando pasamos el cursor por encima

:focus
Cuando el elemento tiene el foco o hacemos click

:active
Cuando está activo

```css
.btn {
    color: red;
}

.btn:visited{
    color: green;
}

.btn:hover{
    color: rgb(235, 10, 138);
}

.btn:focus{
    color: rgb(15, 245, 192);
}

.btn:active{
    color: rgb(252, 96, 13);
}
```

HOVER

Importante:
Las propiedades de nuestro enlace o elemento las definimos en nuestra clase .btn

En la pseudoclase solo pondremos lo que queremos cambiar o añadir respecto a la clase principal.

En el estado hover he cambiado el color de fondo **background-color** y he añadido un redondeo en las esquinas **border-radius**.

El color y el margen (padding) siguen igual que en el estado inicial por lo que **no tengo que volver a ponerlo** en el hover.

```css
.btn{
    color: red;
    background-color: yellow;
    padding:8px 16px;
}

.btn:hover{
    background-color: black;
    border-radius: 12px;
}
```

¿Qué puedo modificar mediante hover ?

- Color
- Background-color
- Padding
- Border
- Text-decoration
- Font-size
- Text-shadow
- Width / height

```css
.btn{
    color: red;
    background-color: yellow;
    padding:8px 16px;
}

.btn:hover{
    background-color: black;
    border-radius: 12px;
}
```

Ejemplo botón

Botón

- **Color**
- **Background-color**
- **Padding**
- **Border**
- **Text-decoration**

Botón

Hover (invertir colores):
Cambiar color fondo
Cambiar color texto
Cambiar color borde

```css
.btn{
    background-color: aquamarine;
    color: rgb(10, 148, 148);
    padding:8px 24px;
    border: 2px solid rgb(10, 148, 148);
    border-radius:25px;
    font-size:24px;
    text-decoration: none;
}

.btn:hover{
    background-color: rgb(10, 148, 148) ;
    color: aquamarine ;
    border: aquamarine ;
}
```

HOVER en otros elementos

Podemos cambiar el estado de cualquier elemento cuando nos situamos en el ratón sobre el (divs, img,texto…):

Ejemplo cambiar anchura de un div con texto.




```
.content{
    background-color: yellow;
    padding:16px;
    width:250px;
}

.content:hover{
    width:400px;
}
```

Opacity

Nos permite cambiar la transparencia de un elemento:

Opacity: 0 transparente
Opacity: 0.1
Opacity: 0.2
Opacity: 0.3
Opacity: 0.4
Opacity: 0.5
Opacity: 0.6
Opacity: 0.7
Opacity: 0.8
Opacity: 0.9
Opacity: 1 opaco

```
.content{
    opacity:0.1;
    background-color: yellow;
    padding:16px;
    width:250px;
}

.content:hover{
    opacity:1;
    width:400px;
}
```

Transform (movimiento, rotación y escala)

Transform nos permite cambiar diferentes aspectos, estos son los más utilizados:

Translate
Nos permite mover elementos

Rotate
Nos permite girar elementos

Scale
Nos permite escalar elementos

Lo aplicamos a:

- divs
- imágenes

con los textos no funciona bien

Link documentación 3wschool

Transform: translate

Podemos mover cada eje (x,y,z) de forma independiente:

Transform:translateX(80px);

Transform:translateY(80px);

Transform:translateZ(80px);

O mover todos los ejes a su vez en una única línea de código separado por comas(el orden sería X-Y-Z):

Transform:translate(80px, 60px);

```
.content:hover{
    transform:translateX(80px);
}

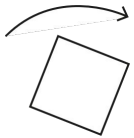
.content:hover{
    transform:translateY(60px;);
}

.content:hover{
    transform:translate3d(80px,40px);
}
```

Transform:rotate

Indicaremos la totación en grados **deg**

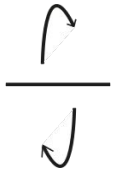
Transform:rotate(80deg);



Transform:translateX(80px);



Transform:translateY(80px);



```
.content:hover{
    transform:rotate(80deg);
}

.content:hover{
    transform:rotateX(80deg);
}


.content:hover{
    transform:rotateY(80deg);
}
```

Transform: scale

Podemos mover cada eje (x,y) de forma independiente:

Transform:scaleX(1.2);

Transform:scaleY(1.2);

O mover todos los ejes a su vez en una única línea de código separado por comas, el orden sería (X-Y):

*En este caso si ponemos un valor afecta a ambos ejes.

Transform:scale(1.2);

Transform:scale(1.2, 0.8);

Indicamos el valor en factor de escalado:

1: es el valor original (100%), por lo que:

1.2 sería aumentar a 120%

2 sería aumentar al 200%

0.8 sería disminuir a 80%

```
.content:hover{
    transform:scale(1.2, 0.8);
}
```

Transition

Nos permite hacer los cambios entre el estado normal y el estado hover de forma gradual.

Indicamos el tiempo que dura la transición en segundo o milisegundo.

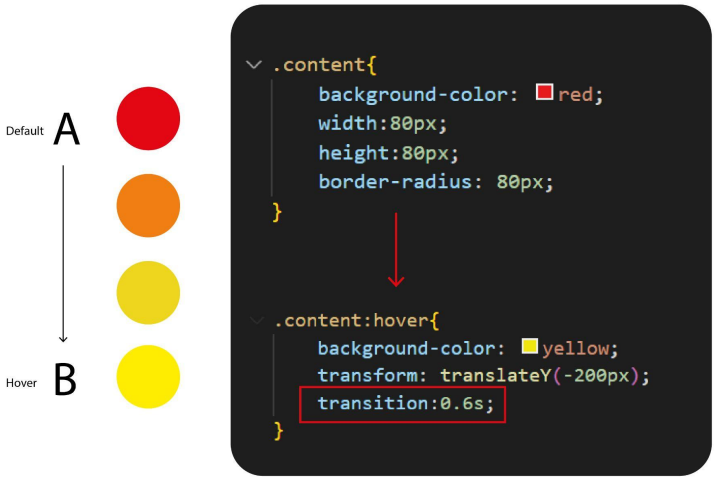
Transition:0.6s;
Transition: 600ms;

```css
.content{
    opacity:0.1;
    background-color: yellow;
    padding:16px;
    width:250px;
}

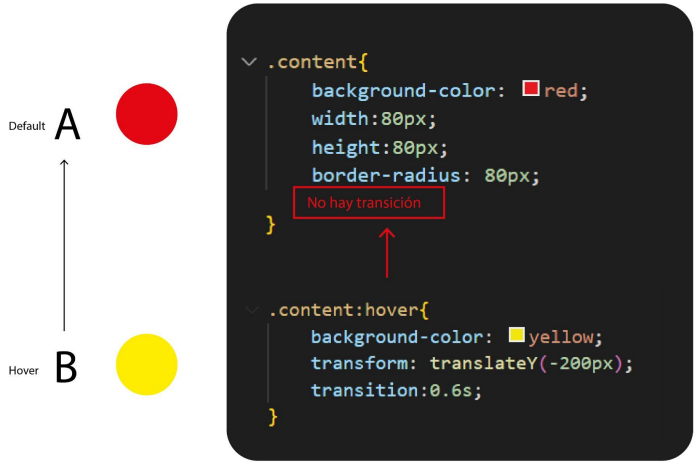
.content:hover{
    transition:0.5s;
    opacity:1;
    width:400px;
}
```

Comportamiento **Transition**

Al hacer **hover** estamos pasando del estado normal (A) al estado hover(B), lo hace de forma gradual gracias al transition que hemos puesto en la pseudoclase :hover.

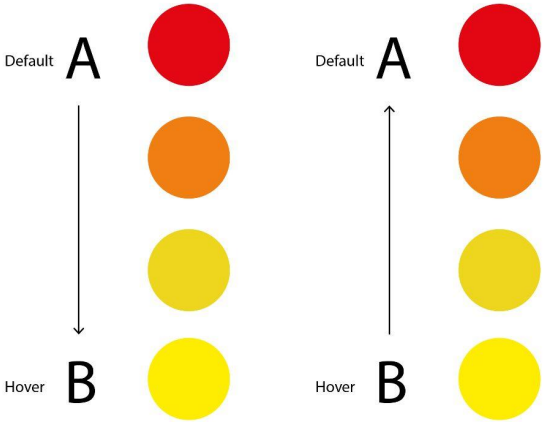


Al dejar de hacer **hover** vamos del estado hover (B) al estado normal(A), **NO** lo hace de forma gradual porque no hemos puesto un transición en la clase original (A).



Comportamiento **Transition**

Si queremos que tanto al hacer hover como al dejar de hacer hover tengamos un cambio gradual, tenemos que poner transition tanto en la **clase (A)** como en la **pseudoclase(B)**.




```css
.content{
    background-color: red;
    width:80px;
    height:80px;
    border-radius: 80px;
    transition:0.6s;
}

.content:hover{
    background-color: yellow;
    transform: translateY(-200px);
    transition:0.6s;
}
```

Transition (aceleración)

La transición tiene un movimiento constante, podemos variar un poco la aceleración con los siguientes valores:

ease
ease-in
ease-out
ease-in-out

Estos valores se indican después del tiempo que dura la transición.

Ver ejmplos

```
.content{
    background-color: red;
    width:80px;
    height:80px;
    border-radius: 80px;
    transition:0.6s ease;
    transition-delay:0.8s;

}
```

Transition-delay

Retardo en que empiece la transición.

Indicamos el tiempo que dura la transición en segundo o milisegundo.

Transition-delay:0.6s;
Transition-delay: 600ms;

```css
.content{
    background-color: red;
    width:80px;
    height:80px;
    border-radius: 80px;
    transition:0.6s;
    transition-delay:0.8s;

}


.content:hover{
    background-color: yellow;
    transform: translateY(-200px);
    transition:0.6s;
}
```

Filtros

Css nos permite aplicar diferentes filtros a imágenes y elementos:

- **Desenfoque**
- **Brillo**
- **Contraste**
- **Sombra(parecido a box-shadow)**
- **Escala de grises**
- **Saturación**
- **Invertir colores**
- **Opacity(hace lo mismo que el vista antes)**
- **Saturación**
- **Tono sepia**

[Link documentación 3wschool]

```
filter: none;
```

```
filter: blur(5px);
```

```
filter: brightness(200%);
```

```
filter: contrast(200%);
```

```
filter: drop-shadow(8px 8px 10px gray);
```

```
filter: grayscale(100%);
```

```
filter: hue-rotate(90deg);
```

```
filter: invert(100%);
```

```
filter: opacity(30%);
```

```
filter: saturate(8);
```

```
filter: sepia(100%);
```

```
filter: contrast(200%) brightness(150%);
```