



# Animaciones con CSS



**Activen las cámaras los que puedan y  
pasemos asistencia**

***Aplicar diseños responsivos  
haciendo uso de media  
queries y breakpoints.***

- Unidad 1: Flexbox.
- Unidad 2: Grid.
- Unidad 3: Media Queries.
- Unidad 4: Animaciones con CSS.



**Te encuentras aquí**





# Inicio

{desafío}  
latam\_



*/\* Crear transiciones con CSS \*/*

*/\* Aplicar transformaciones con CSS. \*/*

*/\* Crear animaciones con CSS. \*/*

# Objetivos

# Activación de conceptos

*Contesta la pregunta correctamente y gana un punto*

## Instrucciones:

- Se realizará una pregunta, el primero en escribir “YO” por el chat, dará su respuesta al resto de la clase.
- El docente validará la respuesta.
- En caso de que no sea correcta, dará la oportunidad a la segunda persona que dijo “Yo”.
- Cada estudiante podrá participar un máximo de 2 veces.
- Al final, el/la docente indicará el 1º, 2º y 3º lugar.
- Esta actividad no es calificada, es solo una dinámica para recordar los conceptos clave para abordar esta sesión.





Activación de conceptos



¿Qué son las media queries y cuál es su función principal en el diseño web?





Activación de conceptos



¿Cuál es la sintaxis básica de una media query?



Activación de conceptos



¿Cuáles son algunos ejemplos comunes de propiedades CSS que se pueden modificar usando media queries?



Activación de conceptos



Primer lugar:

\_\_\_\_\_



Segundo lugar:

\_\_\_\_\_



Tercer lugar:

\_\_\_\_\_



# Desarrollo

{desafío}  
latam\_



# Efectos con CSS

*¿Cómo se logra dar un efecto con CSS?*

En CSS, tenemos 3 propiedades que nos permiten dar efectos a un elemento:

- Transiciones (Transition).
- Transformaciones (Transform).
- Animaciones (Animation).



# Transiciones en CSS

## *Creando una transición*

Con la propiedad **transition** podemos crear efectos de transición suave entre dos estados de un elemento. Veamos un ejemplo:

```
<!-- EL HTML -->
<body>
  <button> Pasa el cursor del
  mouse sobre el botón </button>
</body>
```

```
/* CSS */
button {
  background-color: #0077FF;
  color: white;
  padding: 10px 20px;
  font-size: 16px;
  border: none;
  border-radius: 5px;
  transition: background-color 0.3s ease-out;
}

button:hover {
  background-color: rgb(255, 140, 0);
}
```

# Transiciones en CSS

## Conociendo las transiciones

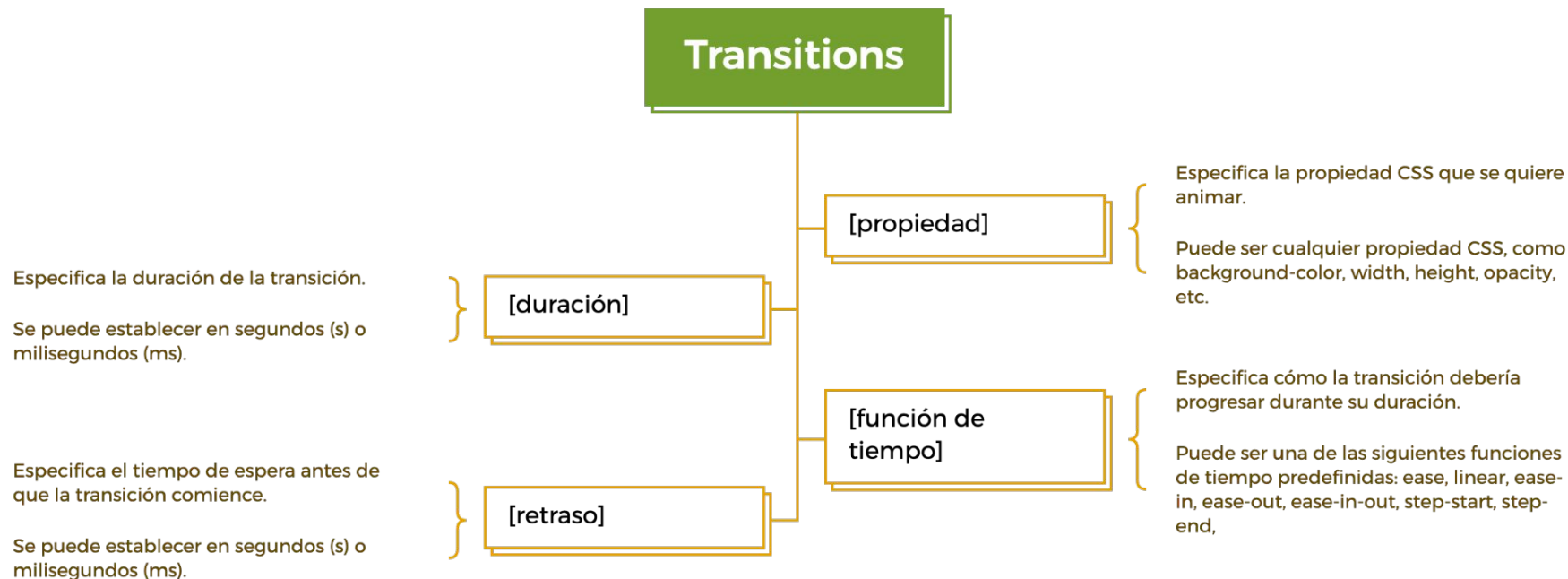
- Para implementar transiciones necesitamos un antes y un después.
- Por ejemplo, un botón que sin hover tiene propiedades con ciertos valores y luego con hover, donde las propiedades tienen un valor distinto.
- La propiedad transition hará que el cambio sea gradual en la cantidad de tiempo especificado.

```
/* CSS */
button {
  background-color: #0077FF;
  color: white;
  padding: 10px 20px;
  font-size: 16px;
  border: none;
  border-radius: 5px;
  transition: background-color 0.3s ease-out;
}

button:hover {
  background-color: rgb(255, 140, 0);
}
```

# Transiciones en CSS

## Componentes del valor transition





# Transiciones en CSS

## *Aplicando transición sobre todas las propiedades*

La transición se puede aplicar al tamaño de una fuente, al tipo de fuente, bordes, margins, paddings, cualquier otra propiedad típica de CSS, y también se puede aplicar a todas las propiedades simultáneamente.

```
.box {  
  width: 100px;  
  height: 100px;  
  background-color: blue;  
  transition: all 1s;  
}  
  
.box:hover {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
}
```

```
<div class="box"></div>
```

# Transiciones en CSS

## *Aplicando transición sobre el ancho*

```
.box {  
  width: 100px;  
  height: 100px;  
  background-color: blue;  
  transition: width 1s;  
}  
  
.box:hover {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
}
```

```
<div class="box"></div>
```



Pregunta



¿Cuál es la diferencia entre aplicar transition a una única propiedad o aplicarlas a todas?

# Transiciones en CSS

## *Cambiando la duración de la transición*

```
.box {  
  width: 100px;  
  height: 100px;  
  background-color: blue;  
  transition: width 1s;  
}
```

```
.box:hover {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
}
```

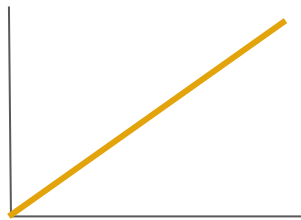
```
<div class="box"></div>
```

← Cambiemos 1s por 3s y observemos la diferencia

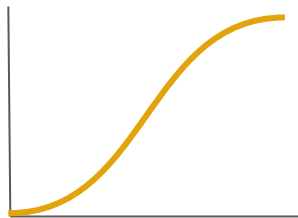
# Transiciones en CSS

## *La función de tiempo*

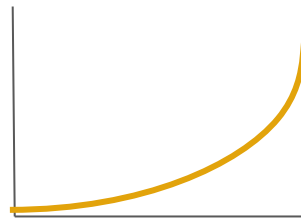
La función de tiempo permite controlar la curva de la velocidad de la transición.



**Linear**  
uniforme



**Ease (por defecto)**  
Lento al principio y  
al final, rápido en la  
mitad



**Ease in**  
Lento al  
principio, rápido  
al final



**Ease out**  
Rápido al principio,  
lento al final

# Transiciones en CSS

## *Aplicando la función de tiempo ease*

```
.box {  
  width: 100px;  
  height: 100px;  
  background-color: blue;  
  transition: all 1s ease;  
}  
  
.box:hover {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
}
```

```
<div class="box"></div>
```

La función ease es por defecto.

# Transiciones en CSS

## Aplicando la función de tiempo linear

```
.box {  
  width: 100px;  
  height: 100px;  
  background-color: blue;  
  transition: all 1s linear;  
}  
  
.box:hover {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
}
```

```
<div class="box"></div>
```

# Transiciones en CSS

## Aplicando la función de tiempo steps

```
.box {  
  width: 100px;  
  height: 100px;  
  background-color: blue;  
  transition: all 1s steps(5);  
}  
  
.box:hover {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
}
```

```
<div class="box"></div>
```



# Transiciones en CSS

## *Tipos de funciones de tiempo*

Función de tiempo	Descripción
ease	Inicia lentamente, luego acelera y luego desacelera hacia el final.
ease-in	Inicia lentamente y luego acelera hacia el final.
ease-out	Inicia rápidamente y luego desacelera hacia el final.
ease-in-out	Inicia lentamente, luego acelera y luego desacelera hacia el final.
linear	Avanza a una velocidad constante durante toda la transición.
step-start	La transición comienza inmediatamente.
step-end	La transición termina inmediatamente.
steps(n)	Divide la transición en n pasos, donde n es un número entero. Cada paso tiene la misma duración y se completa al mismo tiempo.
steps(n, start)	Divide la transición en n pasos, donde n es un número entero. El valor start indica si cada paso comienza o termina en el momento de la transición.

## Ejercicio

En un archivo html nuevo debes crear un link:

- Al hacer hover sobre él, se debe cambiar de color y cambiar la primera letra a mayúscula.
- Puedes utilizar **text-transform: capitalize** para cambiar la primera letra a mayúscula.
- Experimenta con la función de tiempo hasta encontrar un resultado que te guste.
- Retarda la transición 1 segundo.

## Ejercicio

¡Manos al teclado!



*/\* Crear transiciones con CSS \*/* 

*/\* Aplicar transformaciones con CSS. \*/*

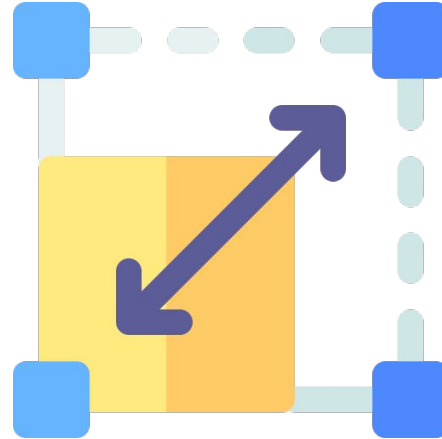
*/\* Crear animaciones con CSS. \*/*

# Objetivos

# Transformaciones

*La propiedad transform*

Las transformaciones en CSS se utilizan para modificar las propiedades geométricas de un elemento, como su tamaño, posición y rotación, utilizando la propiedad **transform**.



# Transformaciones

## *Ejemplo con transform*

```
.box {  
  width: 100px;  
  height: 100px;  
  background-color: blue;  
  transform: rotate(45deg) scale(1.5);  
}
```

```
<div class="box"></div>
```

# Transformaciones

## *Transformaciones posibles*

Transformación	Descripción
<code>translate()</code>	Mueve un elemento en la dirección y distancia especificada.
<code>translateX()</code>	Mueve un elemento horizontalmente en la distancia especificada.
<code>translateY()</code>	Mueve un elemento verticalmente en la distancia especificada.
<code>scale()</code>	Escala un elemento en la cantidad especificada.
<code>scaleX()</code>	Escala horizontalmente un elemento en la cantidad especificada.
<code>scaleY()</code>	Escala verticalmente un elemento en la cantidad especificada.
<code>rotate()</code>	Rota un elemento en el ángulo especificado.
<code>skew()</code>	Sesga un elemento en los ángulos especificados.
<code>skewX()</code>	Sesga horizontalmente un elemento en el ángulo especificado.
<code>skewY()</code>	Sesga verticalmente un elemento en el ángulo especificado.
<code>matrix()</code>	Combina varias transformaciones en una sola matriz.

## Ejercicio

En un archivo html nuevo, debes crear dos cajas de 100px por 100px utilizando `<div class="box">` y asignándole colores por CSS:

- La primera caja debes rotarla 45 grados.
- La segunda caja debes rotarla 30 grados y luego duplicar su tamaño ocupando scale.
- Agrega una transición a una de las cajas, el nuevo tamaño solo debe aplicarse dentro de la transición.

## Ejercicio ¡Manos al teclado!



*/\* Crear transiciones con CSS \*/* ✓

*/\* Aplicar transformaciones con CSS. \*/* ✓

*/\* Crear animaciones con CSS. \*/*

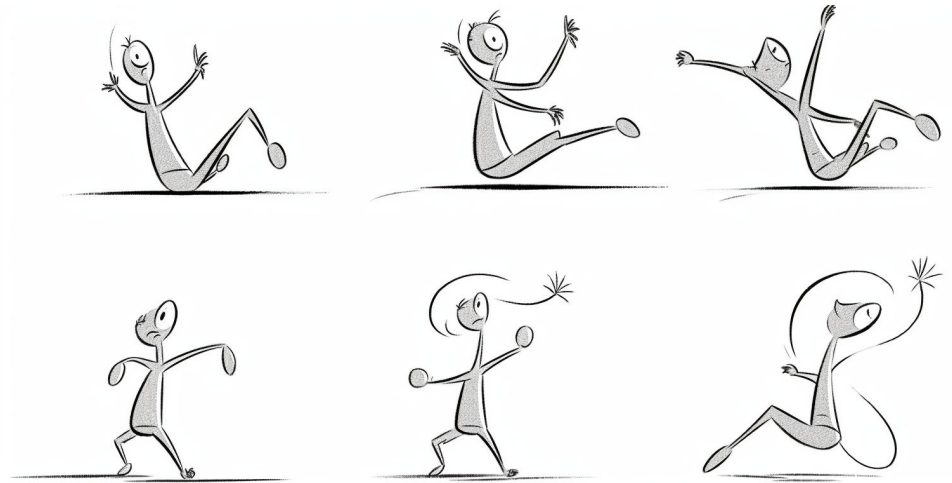
# Objetivos



# Animaciones

## Introducción a Animaciones

- La propiedad "animation" permite crear animaciones más avanzadas y personalizadas mediante la definición de **keyframes**.
- Los keyframes permiten definir los valores intermedios de una animación en diferentes momentos durante su duración.



# Animaciones

## Introducción a Animaciones

Partamos con un ejemplo sencillo, una caja que cambia de color siempre sin necesidad de hover.

```
<div class="box"></div>
```

```
.box {  
  width: 100px;  
  height: 100px;  
  animation-name: colorear;  
  animation-iteration-count:  
infinite;  
  animation-duration: 2s;  
  /* Se aplica la animación  
  "example" a la caja */  
}  
@keyframes colorear {  
  from {  
    background-color: #0079a1;  
  }  
  to {  
    background-color: orange;  
  }  
}
```

# Animaciones


## Conceptos claves

- **animation-name:** El nombre de la animación.
- **animation-iteration-count:** La cantidad de veces que correrá la animación antes de detenerse, también se puede utilizar infinite para un número infinito de veces.
- **@keyframes:** Son los momentos claves de la animación, aquí es donde definimos lo que sucede en cada paso de la animación, utilizamos los keywords from y to para definir el primer y el último paso respectivamente.

# Animaciones

## Ejemplo con texto

```
.container {  
  width: 100%;  
  display: flex;  
  justify-content: center;  
  align-items: center;  
}  
  
.text {  
  font-size: 3em;  
  animation: move 2s infinite;  
}  
  
@keyframes move {  
  from {  
    transform: translateX(0);  
  }  
  to {  
    transform: translateX(200px);  
  }  
}
```



```
<div class="container">  
  <h1 class="text">¡Hola,  
  mundo!</h1>  
</div>
```

# Animaciones

## *Dirección de la animación*

```
.container {  
  width: 100%;  
  display: flex;  
  justify-content: center;  
  align-items: center;  
}  
  
.text {  
  font-size: 3em;  
  animation: move 2s infinite;  
  animation-direction: alternate;  
}  
  
@keyframes move {  
  from {  
    transform: translateX(0);  
  }  
  to {  
    transform: translateX(200px);  
  }  
}
```

```
<div class="container">  
  <h1 class="text">¡Hola,  
  mundo!</h1>  
</div>
```

La propiedad `animation-direction` nos permite hacer una animación reversible y de esta forma lograr una animación más fluida de forma sencilla.

# Animaciones

## Una animación de múltiples etapas

```
.container {  
  width: 100%;  
  height: 100vh;  
  display: flex;  
  justify-content: center;  
  align-items: center;  
}  
  
.text {  
  position: absolute;  
  font-size: 48px;  
  animation: move 4s ease  
infinite;  
}
```

{desafío}  
latam\_

```
@keyframes move {  
  0% {  
    transform: translateX(0);  
  }  
  25% {  
    transform: translateX(100px);  
  }  
  50% {  
    transform: translateX(0);  
    transform: rotate(180deg);  
  }  
  75% {  
    transform:  
translateX(-100px);  
  }  
  100% {  
    transform: translateX(0);  
  }  
}
```

```
<div class="container">  
  <h1  
class="text">¡Hola,  
mundo!</h1>  
</div>
```

## Ejercicio

En un archivo html nuevo, debes crear una caja de 100px por 100px:

- Utilizando keyframes en el instante cero cambia su color a verde.
- En el 50 cambia el color a rojo.
- En el 100 cambia el color azul.
- Haz la animación reversible.

## Ejercicio

¡Manos al teclado!



*/\* Crear transiciones con CSS \*/* ✓

*/\* Aplicar transformaciones con CSS. \*/* ✓

*/\* Crear animaciones con CSS. \*/* ✓

# Objetivos





Cierre

{desafío}  
latam\_



¿Existe algún concepto que no  
hayas comprendido?

Reflexionemos

- Revisar la guía que trabajarán de forma autónoma.
- Revisar en conjunto la prueba.

¿Qué sigue?



*Academia de  
talentos digitales*

[www.desafiolatam.com](http://www.desafiolatam.com)



/DesafioLatam



/DesafioLatam



/DesafioLatam



/DesafioLatam