{desafío}
latam_

SVG, transiciones y animaciones _

Parte II

Transiciones *CSS*

¿Qué son las transiciones de CSS?

- Las transiciones de CSS nos ayudan a definir la duración de un cambio de estado.

- Las transiciones trabajan en base a varias propiedades que controlan diversas variables relacionadas al comienzo y finalización de un estado.

Propiedades de transición

Transition-property

```
.button:hover {
  box-shadow: 10px 10px black;
  transition-property: all;
}
```

- Esta propiedad nos permitirá definir qué propiedades podrán ser afectadas por una transición.
- Existe una gran cantidad de propiedades que podremos afectar.
- Si queremos afectar a todas las propiedades debemos usar el keyword ***all***.

Transition-duration

```
.button:active {
 background-color: gold;
 transition-property: all;
 transition-duration: 1s;
}
```

- Esta propiedad nos permite definir la duración que tendrá la transición de un estado a otro.
- Podemos definir valores en milisegundos **(ms)** y en segundos **(s)**.

Transition-timing-function

```
.button:active {
 background-color: gold;
 transition-property: all;
 transition-duration: 1s;
 transition-timing-function:
ease-in;
}
```

La propiedad ***transition-timing-function*** nos ayudará a definir la aceleración y movimiento que tendrá una transición mediante ***keywords*** con curvas predefinidas o usando una función llamada ***cubic-bezier***().

Función cubic-bezier()

```
transition-timing-function: cubic-bezier(0, .5, 1, 1);
```

Esta función nos permitirá modelar la velocidad que tendrá la transición en función del tiempo definido con la propiedad ***transition-duration.***

Transition-delay

```
.button:hover {
 background-color: gold;
 transition-property: all;
 transition-duration: 1s;
 transition-timing-function:
ease-in;
 transition-delay: 1s;
}
```

- Propiedad que nos permite definir cuánto tiempo de retraso tendrá una transición luego de que un elemento cambie de estado.
- Podemos usar **ms** o **s** para definir esta demora.

Transition

```
.ampolla:hover {
 fill: gold;
 transition: all 1s cubic-bezier(0, .5, 1, 1) 1s;
}
```

- Transition une a todas las propiedades de transición en una sola.
- Los valores a agregar deberán seguir el siguiente orden: **propiedad, duración, velocidad y retardo.**

Animaciones *CSS*

¿Qué son las animaciones de *CSS*?

- Las animaciones nos permitirán dar vida a elementos que antes se encontraban inanimados, tal como si fuera una obra de teatro.

- Para animar un elemento debemos tener en cuenta tres elementos: las **reglas *CSS*** a usar, el **elemento *HTML* o *SVG*** a afectar y los **@keyframes.**

@keyframes

¿Qué son los *@keyframes*?

```
@keyframes opacidad {
 0% {
   opacity: 0;
 }

 60% {
   opacity: .5;
 }

 100% {
   opacity: 1;
 }
}
```

- Son el componente más importante al animar con CSS.
- Con él podremos controlar las etapas que tendrá nuestra animación y definir las reglas que accionaran en la animación.
- Podremos hacer diferentes combinaciones como agregar cambiar el color o desaparecer un elemento.

Propiedades de animación

¿Qué son los *@keyframes*?

```
.button {
 animation-name: mover;
}
```

- Con esta propiedad podremos llamar a la animación creada en un **@keyframes.**

Animation-duration

```
.button {
 animation-name: mover;
 animation-duration: 1s;
}
```

- Esta propiedad nos permitirá definir la duración que tendrá una animación.

- Asimismo podemos usar segundos **(s)** o milisegundos **(ms)** para definir la duración del tiempo.

Animation-timing-function

```
.button {
 animation-name: mover;
 animation-duration: 1s;
 animation-timing-function:
ease-in;
}
```

- Con esta propiedad podemos definir la velocidad que tendrá la animación en función al tiempo definido con ***animation-duration.***

- Para definir la velocidad con la **curva de bézier** o usar los keywords ***ease, linear, ease-in, ease-out*** y ***ease-in-out.***

Animation-delay

```
.button {
 animation-name: mover;
 animation-duration: 1s;
 animation-timing-function:
ease-in;
 animation-delay: 3s;
}
```

- Esta propiedad nos dará la libertad de definir cuándo comenzará la animación, pudiendo comenzar inmediatamente o esperar un momento hasta que comience.

- Los valores que podremos usar son milisegundos **(s)** o segundos **(ms).**

Animation-iteration-count

```
.button {
 animation-name: mover;
 animation-duration: 1s;
 animation-timing-function:
ease-in;
 animation-delay: 3s;
 animation-iteration-count:
3;
}
```

- Esta propiedad es la encargada de definir cuantas veces queremos que se repita una animación.

Animation-direction

```
.button {
 animation-name: mover;
 animation-duration: 1s;
 animation-timing-function:
ease-in;
 animation-delay: 3s;
 animation-iteration-count:
3;
 animation-direction:
alternate;
}
```

- Esta propiedad nos permitirá definir la dirección que tendrá una animación.
- Podemos escoger las siguientes direcciones: ***normal, reverse, alternate o alternate-reverse.***

Animation-fill-mode

```
.button {
 animation-name: mover;
 animation-duration: 1s;
 animation-timing-function:
ease-in;
 animation-iteration-count:
3;
 animation-direction:
alternate;
 animation-fill-mode:
farwards;
}
```

- Propiedad que nos permitirá definir cómo afectarán los estilos de un elemento al comienzo y al final de una animación.
- Podemos mantener los estilos al comienzo ***(backwards),*** al final ***(forwards)*** o en ambos ***(both).***

Animation

```
.button {
 animation: mover 1s ease-in 3 alternate forwards;
}
```

- Al igual que con las transiciones podemos usar una versión reducida en la cual podremos agregar todos los valores de la animación en uno solo.
- El orden que deberá tener los valores es: el nombre del ***@keyframes*, su duración, velocidad, repeticiones, dirección, definición de estilo.**

{desafío} latam_

Academia de talentos digitales

www.desafiolatam.com