


Transiciones de CSS



This is an experimental technology

Because this technology's specification has not stabilized, check the [compatibility table](#) for usage in various browsers. Also note that the syntax and behavior of an experimental technology is subject to change in future versions of browsers as the specification changes.

Las transiciones CSS, parte del borrador de la especificación CSS3, proporcionan una forma de animar los cambios de las propiedades CSS, en lugar de que los cambios surtan efecto de manera instantánea. Por ejemplo, si cambias el color de un elemento de blanco a negro, normalmente el cambio es instantáneo. Al habilitar las transiciones CSS, el cambio sucede en un intervalo de tiempo que puedes especificar, siguiendo una curva de aceleración que puedes personalizar.

 **Nota:** como la especificación de las transiciones CSS todavía se encuentra en fase de borrador, a todas las propiedades asociadas con ellas se les añade el prefijo "-moz-" para usarse en Gecko. Para la compatibilidad con WebKit, se aconseja usar también el prefijo "-webkit-" y para la compatibilidad con Opera, el prefijo "-o-". Es decir, por ejemplo, la propiedad de transición se especificaría como `-moz-transition`, `-webkit-transition` y `-o-transition`.

Las propiedades de transición CSS

Las transiciones CSS se controlan mediante la propiedad abreviada [transition](#). Es preferible utilizar este método porque de esta forma se evita que la longitud de la lista de parámetros sea diferente, lo que puede dar lugar a tener que emplear un tiempo considerablemente largo en depurar el código CSS.

Puedes controlar los componentes individuales de la transición usando las siguientes subpropiedades:

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.

transition-duration

Especifica la duración en la que sucederán las transiciones. Puedes especificar una única duración que se aplique a todas las propiedades durante la transición o valores múltiples que permitan a cada propiedad de transición un período de tiempo diferente.

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.

Detectar la finalización de una transición

Hay un único acontecimiento que se desencadena cuando se completan las transiciones. En Firefox, el evento es `transitionend`, en Opera, `OTransitionEnd` y en WebKit es `webkitTransitionEnd`. Consulta la tabla de compatibilidades al final de la página si deseas más información. El evento `transitionend` ofrece dos propiedades:

propertyName


Una cadena que indica el nombre de la propiedad CSS cuya transición se completó.

elapsedTime

Un float que indica el número de segundos que la transición se había estado ejecutando en el momento en que el acontecimiento se desencadenó. Este valor no está afectado por el valor de `transition-delay`.


Como es habitual, puedes usar el método `element.addEventListener()` para monitorizar este acontecimiento:

```
1 | el.addEventListener("transitionend", updateTransition, true);
```

 **Nota:** el evento "transitionend" no se dispara si la transición se anula debido a que el valor de la propiedad de animación es modificado antes de que la transición se complete.

Propiedades que pueden ser animadas

Las transiciones y las animaciones CSS pueden usarse para animar las siguientes propiedades.

 **Nota:** el conjunto de propiedades que puede animarse está sujeto a cambios, por lo tanto se recomienda evitar incluir cualquier propiedad en la lista que no anime porque en un futuro podría provocar resultados inesperados.

Propiedad	Tipo de valor
<code>background-color</code>	<code><color></code>
<code>background-image</code>	solo degradado; no está implementado en Firefox (see bug 536540)
<code>background-position</code>	<code><percentage></code> <code><length></code>
<code>background-size</code>	<code><percentage></code> <code><length></code>

<code>border-color</code> (including sub-properties)	<code><color></code>
<code>border-radius</code> (including sub-properties)	<code><percentage></code> <code><length></code>
<code>border-width</code> (including sub-properties)	<code><length></code>
<code>border-spacing</code>	<code><length></code>
<code>bottom</code>	<code><percentage></code> <code><length></code>
<code>-moz-box-flex</code>	número
<code>box-shadow</code>	sombra
<code>color</code>	<code><color></code>
<code>-moz-column-count</code>	número
<code>-moz-column-gap</code>	<code><length></code> , palabras clave
<code>-moz-column-rule-color</code>	<code><color></code>
<code>-moz-column-rule-width</code>	<code><length></code> , palabras clave
<code>-moz-column-width</code>	<code><length></code>
<code>clip</code>	rectángulo
<code>fill</code>	pintar
<code>fill-opacity</code>	valor de opacidad
<code>flood-color</code>	<code><color></code> palabras clave
<code>font-size</code>	<code><percentage></code> <code><length></code>
<code>font-size-adjust</code>	números, palabras clave
<code>font-stretch</code>	palabras clave
<code>font-weight</code>	números palabras clave (excluyendo bolder, lighter)
<code>height</code>	<code><percentage></code> <code><length></code>
<code>-moz-image-region</code>	rect()
<code>left</code>	<code><percentage></code> <code><length></code>
<code>letter-spacing</code>	<code><length></code>
<code>lighting-color</code>	<code><color></code> palabras clave
<code>line-height</code>	número <code><percentage></code> <code><length></code>
<code>margin</code> (including sub-properties)	<code><length></code>

marker-offset	<length>
max-height	<percentage> <length>
max-width	<percentage> <length>
min-height	<percentage> <length>
min-width	<percentage> <length>
opacity	número
outline-color	<color>
outline-offset	entero
-moz-outline-radius (including sub-properties)	<percentage> <length>
outline-width	<length>
padding (including sub-properties)	<length>
right	<percentage> <length>
stop-color	<color> palabras clave
stop-opacity	valor de opacidad
stroke	pintar
stroke-dasharray	dasharray
stroke-dashoffset	<percentage> <length>
stroke-miterlimit	miterlimit
stroke-opacity	valor de opacidad
stroke-width	<percentage> <length>
text-indent	<percentage> <length>
text-shadow	sombra
top	<percentage> <length>
-moz-transform-origin	<percentage> <length>, keywords
-moz-transform	transform-function
vertical-align	<percentage> <length>, palabras clave
visibility	visibilidad
width	<percentage> <length>

word-spacing	<percentage> <length>
z-index	entero

Cuando las listas de valores de propiedades tienen longitudes diferentes

Si cualquier lista de valores de propiedades es más corta que las otras, sus valores se repiten para hacer que coincidan. Por ejemplo:

```
1  div {
2    transition-property: opacity, left, top, height;
3    transition-duration: 3s, 5s;
4  }
```

Se considera como si fuera:

```
1  div {
2    transition-property: opacity, left, top, height;
3    transition-duration: 3s, 5s, 3s, 5s;
4  }
```

De manera similar, si cualquier lista de valores de propiedades es más larga que la de **transition-property**, se trunca, de forma que si tienes la siguiente CSS:

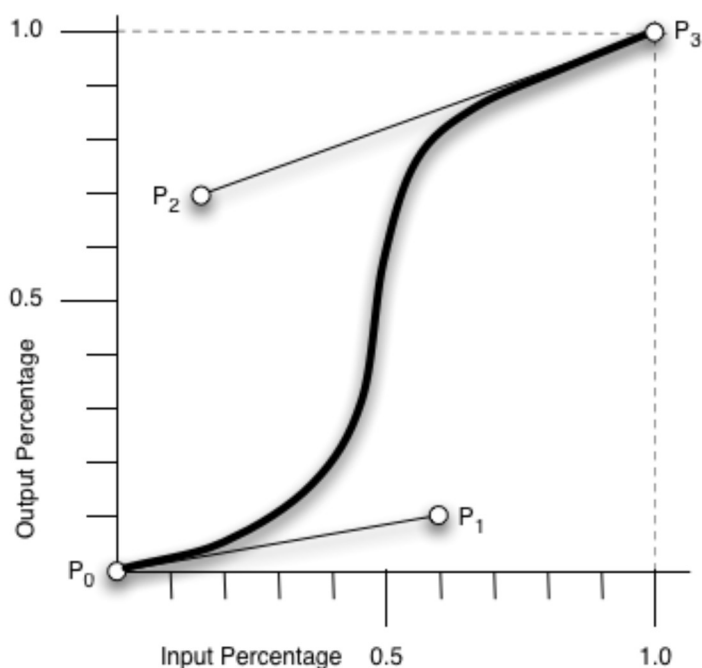
```
1  div {
2    transition-property: opacity, left;
3    transition-duration: 3s, 5s, 2s, 1s;
4  }
```

Se interpreta como:

```
1  div {
2    transition-property: opacity, left;
3    transition-duration: 3s, 5s;
4  }
```

Funciones de intervalos de transición

Las funciones de intervalos determinan el cálculo de los valores intermedios de la transición. La función de intervalo puede especificarse proporcionando el gráfico de la función correspondiente, como lo definen los cuatro puntos que definen una cúbica bézier:



En lugar de especificar directamente una b ezier, existen valores de intervalos predeterminados:

- **ease**, equivalente a `cubic-bezier(0.25, 0.1, 0.25, 1.0)`
- **linear**, equivalente a `cubic-bezier(0.0, 0.0, 1.0, 1.0)`
- **ease-in**, equivalente a `cubic-bezier(0.42, 0, 1.0, 1.0)`
- **ease-out**, equivalente a `cubic-bezier(0, 0, 0.58, 1.0)`
- **ease-in-out**, equivalente a `cubic-bezier(0.42, 0, 0.58, 1.0)`

Ejemplos

Una muestra del efecto de transici n

Este sencillo ejemplo proporciona demostraciones de distintos efectos de transici n sin excesivos adornos.

Antes de que miremos los fragmentos de c digo, tal vez desees [echar un vistazo a la demo en vivo](#) (suponiendo que tu navegador admita transiciones). Tambi n puedes echar un [vistazo directamente a la CSS](#) que usa.

En primer lugar, el HTML para crear los elementos sobre los que probaremos nuestras transiciones:

```
1 <ul>
2   <li id="long1">Transici n larga, gradual...</li>
3   <li id="fast1">Transici n muy r pida...</li>
4   <li id="delay1">Transici n larga de un minuto de retraso...</li>
5   <li id="easeout">Usar intervalos de alejamiento...</li>
6   <li id="linear">Usar intervalos lineales...</li>
7   <li id="cubic1">Usar c bica b ezier(0.2, 0.4, 0.7, 0.8)...</li>
8
```

```
</ul>
```

Cada elemento tiene su propia id.; la CSS se encarga del resto. Veamos un par de ejemplos.

Usar un retraso

Este ejemplo realiza una transición de tamaño de fuente de cuatro segundos con dos segundos de retraso entre el momento en que el usuario pasa el ratón por encima del elemento y el comienzo del efecto de animación:

```
1  #delay1 {
2    position: relative;
3    transition-property: font-size;
4    transition-duration: 4s;
5    transition-delay: 2s;
6    font-size: 14px;
7  }
8
9  #delay1:hover {
10   transition-property: font-size;
11   transition-duration: 4s;
12   transition-delay: 2s;
13   font-size: 36px;
14 }
```

Usar una función de intervalos de transición lineales

De manera predeterminada, la función de intervalos que se usa para computar los pasos intermedios durante la secuencia de animación proporciona una curva suave de aceleración y desaceleración para el efecto de animación. Si prefieres que el efecto mantenga una velocidad constante a lo largo de la animación, puedes especificar que deseas usar la función de intervalos de transición `linear`, tal y como se muestra a continuación.

```
1  transition-timing-function: linear;
```

Existen distintas funciones de intervalos estándares disponibles; consulta [transition-timing-function](#) para tener más detalles.

Especificar una función de intervalos cúbicos bézier

Puedes controlar aún más el intervalo de la secuencia de animación si especificas tu propia curva cúbica bézier que describe la velocidad de animación. Por ejemplo:

```
1  transition-timing-function: cubic-bezier(0.2, 0.4, 0.7, 0.8);
```


Establece una función de intervalo con una curva bézier definida por los puntos (0.0, 0.0), (0.2, 0.4), (0.7, 0.8) y (1.0, 1.0).

Menús de resaltado

Un uso común de CSS es resaltar elementos de un menú mientras el usuario desplaza el cursor del ratón por encima de ellos. Es fácil usar las transiciones para hacer que el efecto sea aún más atractivo.

Antes de que miremos los fragmentos de código, tal vez desees [echar un vistazo a la demo en vivo](#) (suponiendo que tu navegador admita transiciones). También puedes echar un [vistazo directamente a la CSS](#) que usa.

Primero configuramos el menú usando HTML:

```
1 <div class="sidebar">
2   <p><a class="menuButton" href="home">Inicio</a></p>
3   <p><a class="menuButton" href="about">Acerca de</a></p>
4   <p><a class="menuButton" href="contact">Contacto Us</a></p>
5   <p><a class="menuButton" href="links">Vínculos</a></p>
6 </div>
```

Después construimos la CSS para implementar el aspecto de nuestro menú. Las porciones relevantes se muestran a continuación:

```
1 .menuButton {
2   position: relative;
3   transition-property: background-color, color;
4   transition-duration: 1s;
5   transition-timing-function: ease-out;
6   -webkit-transition-property: background-color, color;
7   -webkit-transition-duration: 1s;
8   -o-transition-property: background-color, color;
9   -o-transition-duration: 1s;
10  text-align: left;
11  background-color: grey;
12  left: 5px;
13  top: 5px;
14  height: 26px;
15  color: white;
16  border-color: black;
17  font-family: sans-serif;
18  font-size: 20px;
19  text-decoration: none;
20  -moz-box-shadow: 2px 2px 1px black;
21  padding: 2px 4px;
22  border: solid 1px black;
23 }
```



```
}

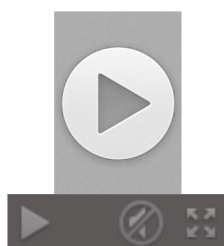
.menuButton:hover {
  position: relative;
  transition-property: background-color, color;
  transition-duration: 1s;
  transition-timing-function: ease-out;
  -webkit-transition-property: background-color, color;
  -webkit-transition-duration: 1s;
  -o-transition-property: background-color, color;
  -o-transition-duration: 1s;
  background-color:white;
  color:black;
  -moz-box-shadow: 2px 2px 1px black;
}
```

Esta CSS establece el aspecto del menú con los colores de fondo y del texto que cambian cuando el elemento está en su estado `:hover`.

En lugar de describir el efecto con todo detalle, puedes echar un [vistazo a la muestra en vivo](#) si tu navegador admite transiciones (Firefox y WebKit nightlies, Opera 10.5).

Usar eventos de transición para animar un objeto

En este ejemplo, una pequeña caja con texto dentro se mueve hacia atrás y hacia delante a través de la pantalla y los colores de fondo y del texto se difuminan entre dos valores mientras tiene lugar la animación.



Antes de que miremos los fragmentos de código, tal vez desees [echar un vistazo a la demo en vivo](#) (suponiendo que tu navegador admita transiciones). También puedes echar un [vistazo directamente a la CSS](#) que usa.

El HTML

El HTML para este ejemplo es muy sencillo:

```
1 | <!DOCTYPE html>
2 | <html>
3 |
```

```
<head>
  <title>CSS Transition Demo</title>
  <link rel="stylesheet" href="transitions.css" type="text/css">
  <script src="transitions.js" type="text/javascript"></script>
</head>
<body onload="runDemo()">
  <div class="slideRight">¡Esto es una caja!</div>
</body>
</html>
```

Lo único que hay que observar aquí es que establecemos la clase para nuestra caja en "slideRight" inicialmente y cuando el documento haya terminado de cargarse, se ejecuta la función `runDemo()` del código JavaScript.

La CSS

Para crear nuestro efecto de animación, usamos dos clases de CSS, "slideRight" y "slideLeft". Si deseas ver el código completo de CSS, puedes mirar el archivo [transitions.css](#) en su totalidad. A continuación se muestran sólo los trozos relevantes:

```
1  .slideRight {
2    position: absolute;
3    transition-property: background-color, color, left;
4    transition-duration: 5s;
5    -webkit-transition-property: background-color, color, left;
6    -webkit-transition-duration: 5s;
7    -o-transition-property: background-color, color, left;
8    -o-transition-duration: 5s;
9    background-color: red;
10   left: 0%;
11   color: black;
12 }
```

Observa que aquí especificamos de manera explícita la propiedad de posición. Esto es necesario porque sólo aquellos elementos cuya propiedad de posición se defina de manera expresa pueden animar su posición.

La propiedad **transition-property** se usa para enumerar las propiedades CSS que deseamos animar. En este caso, las propiedades que se van a animar son **background-color**, **color** y **left**. La propiedad **transition-duration** indica que deseamos que la animación tarde 5 segundos desde que comienza hasta que termina.

Se incluyen los equivalentes WebKit y Opera para permitir que el ejemplo funcione en el software correspondiente.

La clase "slideRight" se usa para especificar el punto de inicio para que la animación desplace el elemento

desde el borde izquierdo hasta el borde derecho de la ventana del navegador. Como tal, define la posición y el color del elemento cuando está al principio de la secuencia de animación; concretamente, el valor para su propiedad `left` es 0%, lo que indica que comenzará en el borde izquierdo de la ventana.


Se muestra a continuación la clase "slideLeft", que define el punto final de la animación, es decir, el punto en el que concluirá la animación de izquierda a derecha y cambiaremos a una animación de derecha a izquierda.

```
1  .slideLeft {  
2    position: absolute;  
3    transition-property: background-color, color, left;  
4    transition-duration: 5s;  
5    -webkit-transition-property: background-color, color, left;  
6    -webkit-transition-duration: 5s;  
7    -o-transition-property: background-color, color, left;  
8    -o-transition-duration: 5s;  
9    text-align: center;  
10   background-color: blue;  
11   left: 90%;  
12   color: white;  
13   width: 100px;  
14   height: 100px;  
15 }
```

Los valores de color aquí se han cambiado para hacer que los colores de fondo y del texto cambien durante el tiempo de la secuencia de animación. Además de esto, la propiedad `left` está aquí al 90%.

El código JavaScript

Una vez que hemos establecido los extremos de la secuencia de animación, lo que tenemos que hacer es iniciar la animación. Podemos hacerlo fácilmente usando JavaScript.

 **Nota:** una vez que [la compatibilidad para las animaciones CSS](#) esté disponible, el código JavaScript no será necesario para lograr este efecto.

En primer lugar, la función `runDemo()` que se llama cuando el documento se carga para inicializar la secuencia de animación:

```
1  function runDemo() {  
2    var el = updateTransition();  
3  
4    // Configurar un controlador de eventos para invertir la dirección  
5    // cuando finalice la transición.  
6  
7    el.addEventListener("transitionend", updateTransition, true);  
8  }
```

```
}
```

Es bastante sencillo: llama a la función `updateTranslation()` que definiremos enseguida, cuyo trabajo es establecer la clase para el elemento que estamos animando según la dirección en la que queramos que viaje. A continuación configura un proceso de escucha de evento para observar el evento "transitionend" que se envía cuando se completa una transición; esto nos permite saber cuándo es el momento para cambiar la clase del elemento para revertir la dirección de la animación.

La función `updateTransition()` tiene este aspecto:

```
1 function updateTransition() {  
2     var el = document.querySelector("div.slideLeft");  
3  
4     if (el) {  
5         el.className = "slideRight";  
6     } else {  
7         el = document.querySelector("div.slideRight");  
8         el.className = "slideLeft";  
9     }  
10  
11     return el;  
12 }
```

Esto ubica el elemento que estamos animando al buscarlo por su nombre de clase (aquí podríamos usar una id, por supuesto, pero seguidme la corriente). En primer lugar buscamos el nombre de la clase "slideLeft". Si se encuentra, cambiamos la clase del elemento a "slideRight". Esto iniciará la transición de derecha a izquierda, puesto que es el momento de que se deslice a la izquierda si el elemento está ya en el borde derecho, que será cuando llegue el evento "transitionend" y la clase del elemento sea "slideLeft" (se deslice a la izquierda).

Si no se halla ningún elemento que coincida con la clase "slideLeft", buscamos el elemento que coincida con "slideRight" y cambiamos su clase a "slideLeft", comenzando de ese modo la animación en la dirección contraria.

Compatibilidad de navegadores

Navegadores	Compatibilidad básica	Propiedad	Evento de transición finalizada
Internet Explorer	(ninguna, a partir de IE9 pp7)	---	---
Firefox (Gecko)	4.0 (2.0)	-moz-transition	transitionend
Opera	10.5	-o-transition	OTransitionEnd

Safari Chrome WebKit	3.2 yes yes	-webkit-transition	webkitTransitionEnd
--------------------------	-----------------	--------------------	---------------------

Compatibilidad de navegadores

Desktop

Mobile

Funcionalidad Chrome Firefox (Gecko) Internet Explorer Opera Safari (WebKit)

Compatibilidad básica ? ? ? ? ?

Consultar también

- [🔗 Módulo de transiciones CSS nivel 3](#)
- `-moz-transition`
- `-moz-transition-property`
- `-moz-transition-duration`
- `-moz-transition-timing-function`
- `-moz-transition-delay`

HTML5 Documentation

HTML	Audio/Video Canvas WebGL SVG MathML	WebForms AppCache Microformats	SemanticTags
JavaScript	Storage IndexedDB WebSockets	WebWorkers Events Drag/Drop ProtocolHandler	Geolocation Focus
CSS	NewSelectors	Typography	Visual Effects

Aprender lo mejor del desarrollo web



Subscríbete a nuestro boletín:

SUSCRIBIRSE

Por ahora, el boletín solo está disponible en inglés.