Animación con Keyframes en CSS3

Seguro has visto animaciones impresionantes con CSS3, quizás las más vistosas usaban efectos tridimensionales (antes de entrar en ese área conviene manejar muy bien los bidimensionales) y quizás cuando hayas visto el código que se necesita para provocar esas animaciones te sorprenda que sea tan breve en la mayoría de los casos, y a la vez tan sencillo.

Solo necesitas comprender bien qué hace cada uno de los diferentes efectos que se aplican y saber combinarlos unos con otros. Si tienes creatividad y habilidad para plasmar en código lo que imaginas en tu mente, disfrutarás mucho creando tus propias animaciones CSS3.

Pero antes veamos los conceptos fundamentales de cualquier animación keyframe

Cómo hacer una animación usando Keyframes en CSS3

Una animación usando keyframes se compone de 3 valores:

- **el nombre** de la animación,
- **la duración** que tendrá y
- **la cantidad** de veces que se reproducirá.

Además hay que pensar bien dónde hacer la llamada, es decir, en qué situación se reproducirá la animación.

> *En los casos que se explicarán en este artículo las animaciones se harán sobre divs cuando el ratón se pose encima, en el :hover. Se usarán de forma orientativa los prefijos moz y webkit.*

El código correspondiente para crear una animación sería el siguiente:

Código :

```
.caja_animada:hover{

  -moz-animation-name: nombre_animacion;
  -moz-animation-duration: 2s;
  -moz-animation-iteration-count: 1;

  -webkit-animation-name: nombre_animacion;
  -webkit-animation-duration: 2s;
  -webkit-animation-iteration-count: 1;

  /*para que se reproduzca constantemente pondríamos el valor
infinite en animation-iteration-count */
}
```

Sin embargo se pueden simplificar todos los parámetos de la animación en una sola línea de la siguiente forma:

Código :

```
.caja_animada:hover{
  -webkit-animation: 2s nombre_animacion 1;
  -moz-animation: 2s nombre_animacion 1;
}
```

Estructura y funcionamiento de un Keyframe

Una vez visto cómo invocar un keyframe, veamos cómo funciona la estructura de éstos para poder desarrollar la animación:

Los Keyframes se desarrollan indicando qué sucederá en el objeto a medida que transcurre el tiempo en la animación, y el tiempo se indica con los porcentajes.

Código :

```
@-moz-keyframes nombre_animacion{
  20% {
    /* código css 1 */
  }

  40%, 80% {
    /* código css 2 */
  }

  65% {
    /* código css 3 */
  }
}
```

- Si el tiempo de duración de la animación es de 1 segundo, el 20% de la animación correspondería a las dos décimas de iniciar la animación.

- Si queremos que cierto efecto suceda en la mitad de la animación tendríamos que codearlo entre las llaves del 50% que deberíamos indicar.

- Cuando queremos que nuestra animación acabe igual que comenzó, no es necesario poner los atributos css iniciales en el punto 100%, ya que **por defecto las animaciones una vez concluidas regresan al estado inicial.**

También es interesante notar que **podemos reutilizar código,** si queremos que ciertas características se reproduzcan de forma idéntica en varios puntos de la animación **podemos ponerlos todos juntos separados de comas** como se ven en el ejemplo.

Una vez inicie la animación, ésta no se reproducirá en la misma secuencia que hayamos escrito las líneas. Es decir tomando como ejemplo el código de arriba, la secuencia no será:

- código css 1
- código css 2
- código css 3

Sino que **irá en orden en el tiempo que hayamos indicado a través de los porcentajes,** es decir, la secuencia será:

- código css 1 (al 20% de la animación)
- código css 2 (al 40% de la animación)
- código css 3 (al 65% de la animación)
- código css 2 (al 80% de la animación)

Así que **nuestra línea del tiempo en la animación va marcada por los porcentajes** y la secuencia irá de forma progresiva avanzando desde los porcentajes menores a los más mayores, aunque ésto implique que el código a ejecutar se encuentre antes o después, como se ve en ejemplo, después de hacer el código 3, regresa arriba a hacer el código 2. ¿Por qué? Porque el código 2 que también tiene que reproducirse en el 80% lo tiene que hacer después del 65%, por eso regresa arriba, ya que se guía por orden en el tiempo establecido por los porcentajes.

Una vez teniendo estos conceptos en mente, veamos **ejemplos de animaciones**

que podemos hacer con keyframes. Les animo que para comprender mejor estas animaciones las codeen y vean el resultado directamente en su navegador.

Efecto Rebote

En este ejemplo conseguimos que una caja se eleve 2 veces como si estuviera botando, notemos que la primera elevación se hace al 40% de 30px hacia arriba, después regresa abajo, y vuelve a elevarse pero ésta vez menos, 15px y vuelve a bajar. También podemos ver que el tiempo de duración entre elevación y elevación no es el mismo, el primer rebote se hace en una transición del 20 al 40 % y la segunda en una transición más breve, del 50 al 65 % dando la sensación de que el primer rebote es más lento.

Con estas combinaciones se logran simular los rebotes reales de un elemento, ya que cuando, por ejemplo una pelota se lanza y rebota contra la superficie, cada vez los rebotes son más pequeños.

Código :

```
.caja_animada:hover{
  -moz-animation: 2s bote 1;
}

@-moz-keyframes bote {
  20%, 50%, 80% {
    -moz-transform: translateY(0);
  }

  40% {
    -moz-transform: translateY(-30px);
  }

  65% {
    -moz-transform: translateY(-15px);
  }
}
```

Mira el resultado en el video:

Efecto Intermitente

En este ejemplo conseguimos que una caja haga una intermitencia doble, o un efecto flash, desapareciendo 2 veces, al 25 y 75%, por eso necesitamos en mitad de esas desapariciones que vuelva a verse completamente (opacity: 1 al 50%) para que la segunda desaparición tenga sentido.

Código :

```
.caja_animada:hover{
   -webkit-animation: 2s intermitente 1;
}

@-webkit-keyframes intermitente {
  25%, 75% {
    opacity: 0;
  }
  50% {
    opacity: 1;
  }
}
```

Mira el resultado en el video:

Efecto Doble Click

En este caso lo que se logra es que una caja cambie su tamaño dos veces simulando un doble click sobre ella, se reduce en dos ocasiones y entre ambas se aumenta más de lo inicial para resaltar el efecto.

Código :

```
.caja_animada:hover{
    -webkit-animation: 1s dobleclick 1;
}

@-webkit-keyframes dobleclick {
  40% {
    -webkit-transform: scale(1.1);
  }
  20%, 60% {
    -webkit-transform: scale(0.8);
  }
}
```

Mira el resultado en el video:

Efecto Chicle

En esta ocasión nos valemos de las transformaciones de escalas tratando las anchuras y alturas por separado. Primero estiramos la caja a los lados y la achicamos en altura, luego hace el efecto de rebotar y se estrecha aumentando algo el tamaño y volviendo a estirarse de los lados hasta volver a su estado inicial. Como si fuera un chicle, parecido también a un muelle.

Código :

```
.caja_animada:hover{
    -webkit-animation: 2s chicle 1;
}

@-webkit-keyframes chicle {
  30% {
    -webkit-transform: scaleX(1.25) scaleY(0.75);
  }

  40% {
    -webkit-transform: scaleX(0.75) scaleY(1.25);
  }

  60% {
    -webkit-transform: scaleX(1.15) scaleY(0.85);
  }
}
```

Mira el resultado en el video:

Efecto Balaceo

Con este ejemplo provocamos diferentes sensaciones, por un lado al inicio, al reducir al tamaño mientras que se rota da la impresión de que la caja se ha ido hacia atrás, y después al aumentar su tamaño parece que se viene hacia nosotros, girando varias veces de un lado a otro provocando el balanceo propio.

Código :

```
.caja_animada:hover{
   -moz-animation: 2s balanceo 1;
}

@-moz-keyframes balanceo {

  20% {
    -moz-transform: scale(0.9) rotate(-3deg);
  }

  30%, 50%, 70%, 90% {
    -moz-transform: scale(1.1) rotate(3deg);
  }

  40%, 60%, 80% {
    -moz-transform: scale(1.1) rotate(-3deg);
  }
}
```