Diseño de Interfaces Web

UD – 5

Animaciones

Contenidos

ÍNDICE

1. INTRODUCCIÓN

En este tema vamos a hablar de 3 cosas distintas:

- **Transformaciones** → *Es simplemente aplicar alguna transformación espacial a un elemento HTML como moverlo, rotarlo o agrandarlo.*
- **Transiciones** → *Es un API de CSS para hacer que las cosas se muevan en la pantalla.*
- **Animaciones** → *Es otro API CSS mas complejo hacer que las cosas se muevan en la pantalla.*

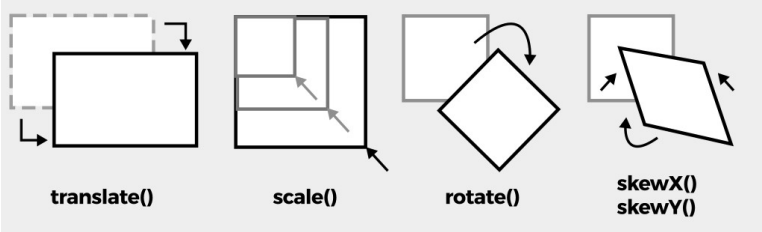
¿Que podemos hacer con animaciones?

- ***404***
- ***Reloj***
- ***Escaleras***
- ***Cuadros animados***
- ***Monument Valley***

Lo mas importante de las animaciones es usarlas cuando tengan sentido y no solo como una moda o capricho. Es decir hay que usarlas cuando aportan algo al usuario.

2. TRANSFORMACIONES

Las transformaciones se pueden efectuar en CSS mediante la propiedad transform que permite recibir una función de transformación determinada, la cuál será aplicada en el elemento HTML en cuestión seleccionado mediante CSS. Permitiendo realizar acciones del tipo:



TRANSFORMACIONES

ESCALADO

```
selector {
    //En ambas dimensiones
    transform: scale(valueX,valueY);
    //Sólo en X
    transform: scaleX(valueX);
    //Sólo en Y
    transform: scaleY(valueY);

}
```

TRANSFORMACIONES

TRANSLACIÓN

```
selector {
    //En ambas dimensiones
    transform: translate(valueX,valueY);
    //Sólo en X
    transform: translateX(valueX);
    //Sólo en Y
    transform: translateY(valueY);

}
```

ROTACIÓN

```
selector {

    transform: rotate(angulo);

}

// Expresaremos los ángulos como (45deg) o (-45deg)
// deg por degrees. Se pueden usar otras unidades
// En el sentido de las aguja del reloj
```

TRANSFORMACIONES

SESGADOS

```
selector {
    //En ambas dimensiones
    //No funciona en algunos navegadores
    transform: skew(anguloX,anguloY);
    //Sólo en X
    transform: skewX(anguloX);
    //Sólo en Y
    transform: skewY(anguloY);
}
```

Se pueden emplear múltiples transformaciones separándolas mediante espacio. En el siguiente ejemplo, aplicamos una función de rotación, una función de escalado y una función de traslación de forma simultánea:

```
.img {
  transform: rotate(5deg) scale(2) translate(25px, 150px);
}
```

3. TRANSICIONES

Las transiciones se usan sobre todo para hacer cambios en los eventos de :

- :hover
- :focus
- etc...

La forma concreta de la propiedad transition es:

transition: propiedadCSS duracion timing-function retardo

- ✔ **propiedadCSS →** La propiedad CSS a la que se le aplica la transición.
- ✔ **duracion →**Duración en segundos de la transición
- ✔ **timing-function** La velocidad de la transición. Sus valores son:
  - *linear: Velocidad constante.*
  - *ease*
  - *ease-in: Aceleración constante.*
  - *ease-out*
  - *ease-in-out*
  - *steps(pasos)*
  - *cubic-bezier(n,n,n,n)*
- ✔ **retardo**: Cuanto tiempo en segundos tarda en empezar la transición.

Un ejemplo aclaratorio sería:

```
.c-button {
        padding: 10px 20px 10px 20px;
        background-color: #ffffff;
        border: 1px solid #bbd4f7;
        border-radius: 4px;
        cursor: pointer;
        transition: background-color 1s, box-shadow 1s;
 }

         .c-button:hover {
           background-color: #deebfc;
           box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2);
 }
```

4. ANIMACIONES

Las animaciones son transiciones entre elementos que se definen mediante reglas de estilo en CSS. Se distinguen dos partes fundamentales en la creación de dichas animaciones:

- *Los fotogramas, que indican el estado inicial y final de estas.* → @keyframes
- *El estilo CSS, que define la transición de la animación.* → animation-*

La propiedad **"animation"** presenta diversos parámetros que permiten configurar su funcionamiento. Cabe mencionar que, con esta propiedad, solo se determinan las propiedades de la animación, pero la apariencia de esta se define a través de **@keyframes**. Los parámetros más habituales son:

Propiedades de animación

Para el comportamiento de una animación, necesitamos conocer las siguientes propiedades, que son una «ampliación» de las propiedades de las transiciones CSS:

Propiedades	Descripción	Valor
animation-name	Nombre de la animación a aplicar.	**none** | *nombre*
animation-duration	Duración de la animación.	**0** | TIME
animation-timing-function	Ritmo de la animación.	Ver funciones de tiempo
animation-delay	Retardo en iniciar la animación.	**0** | TIME
animation-iteration-count	Número de veces que se repetirá.	**1** | infinite | NUMBER
animation-direction	Dirección de la animación.	**normal** | reverse | alternate | alternate-reverse
animation-fill-mode	Como se «completa» la animación.	**none** | forwards | backwards | both
animation-play-state	Estado de la animación.	**running** | paused

Atajo de animaciones

CSS ofrece la posibilidad de resumir todas estas propiedades en una sola, para hacer nuestras hojas de estilos más compactas. El orden recomendado para los valores de la propiedad de atajo sería el siguiente:

```
div {
  /* animation: <name> <duration> <timing-function> <delay>
                <iteration-count> <direction> <fill-mode> <play-state> */

  animation: change-color 5s linear 0.5s 4 normal forwards running;
}
```

Animaciones múltiples

es posible separar por comas para indicar múltiples valores, en este caso, para indicar que queremos realizar varias animaciones a la vez:

```
.img {
        animation:
                move-right   5s   linear,
                change-color   5s   linear;
}
```

Aclaración: *, indicamos que tanto la animación move-right como la animación change-color deben empezar a la vez, cada una de las cuales dura 5 segundos y tienen un ritmo lineal (constante):*

Encadenar animaciones

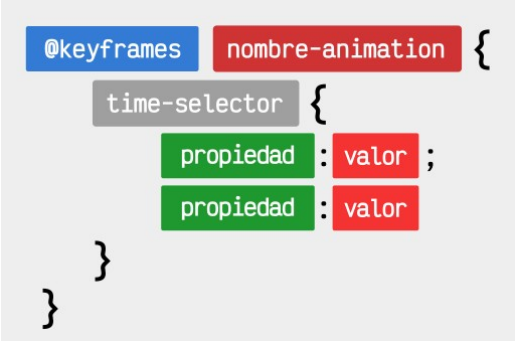
Es posible encadenar animaciones utilizando animaciones múltiples combinadas con la propiedad animation-delay.

```
img {
    animation:
        move-right   5s linear 0s,        /* Comienza a los 0s (no hay anterior) */
        look-up   2.5s linear 5s,         /* Comienza a los 5s (5 de la anterior) */
        move-left   2s linear 7.5s,       /* Comienza a los 7.5s (5 + 2.5 de anterior) */
        dissapear   2s linear 9.5s;       /* Comienza a los 9.5s (5 + 2.5 + 2 anterior) */
}
```

Aclaración: *El truco de este ejemplo está en los valores de duración y los de retardo, en combinación con los anteriores:*

Keyframes : Fotogramas

Para definir esos fotogramas clave, utilizaremos la regla **@keyframes** y se basa en el siguiente esquema :



La descripción de los diferentes campos, aparece desglosada en la siguiente tabla:

Parte	Descripción
`@keyframes` STRING *name*	Regla para darle un nombre y definir los fotogramas clave de una animación.
`from`	Fotograma clave inicial con los estilos CSS a aplicar. Equivalente a `0%`.
`to`	Fotograma clave final con los estilos CSS a aplicar. Equivalente a `100%`.
PERCENT	Porcentaje específico de la animación con los estilos CSS a aplicar. Permite decimales.

Un ejemplo descriptivo final de las animaciones sería:

```
@keyframes cambiar-color {

    0% {
        background: red;                /* Primer fotograma */
    }
    50% {
        background: yellow;             /* Segundo fotograma */
        width: 400px;
    }
    100% {
        background: green;              /* Último fotograma */
    }
}

.img {
    background: grey;
    color: #FFF;
    width: 150px;
    height: 150px;
    animation: cambiar-color 2s ease 0s infinite;
}
```