TRANSFORMACIONES - TRANSICIONES - ANIMACIONES EN CSS

Contenido

Con este documento se pretende realizar un breve resumen y analizar algunos ejemplos donde veamos la importancia de CSS y como con una pocas líneas de código se pueden realizar efectos muy visuales. El documento se divide en tres apartados: transformaciones, transiciones y animaciones.

1. TRANSFORMACIONES CON CSS.

Las transformaciones en CSS nos permitirán modificar el comportamiento de un elemento, así podremos realizar transformaciones en 2D y 3D. Para poder realizar una transformación utilizaremos la propiedad transform.

Algunas de las operaciones de transformación que podemos realizar a un objeto son:

Translación: Traslada un elemento. Esta traslación puede realizarse, en el eje x, en el eje y o en ambos a la vez.

- translateX(valorx)
- translateY(valory)
- translate(valorx,valory)

Escalado: Escala un elemento. Este escalado puede realizarse en el eje x, en el eje y o en ambos. Como parámetro indicamos cuanto queremos escalar nuestro elemento. Así un valor de 2 significa que queremos doblar el tamaño de nuestro objeto.

- scaleX(aumentoX)
- scaleY(aumentoY)
- scale(aumentoX,aumentoY)

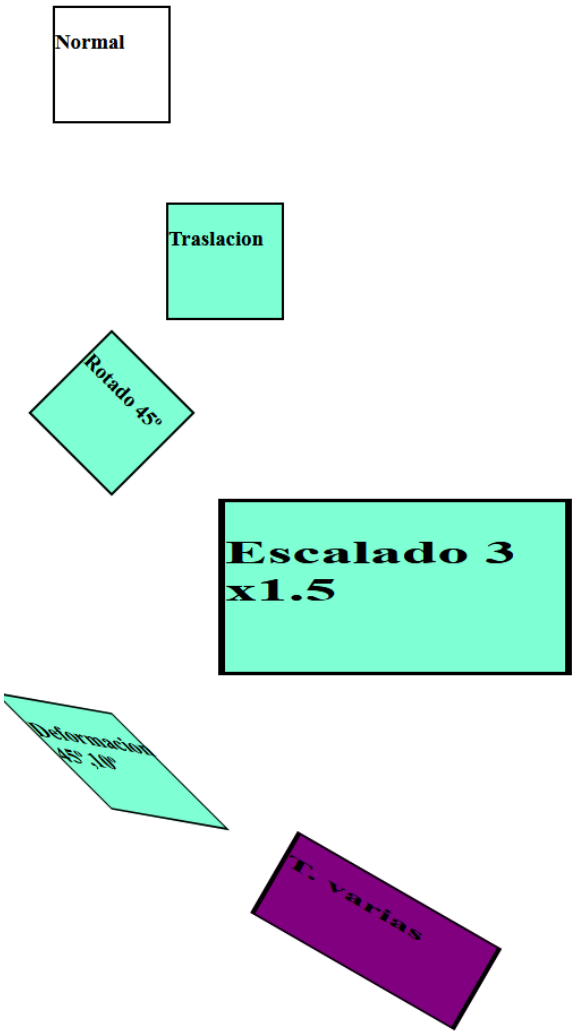
Rotación: Permite girar un elemento un determinado número de grados.

- rotate(grados)

Deformación: Permite aplicar una deformación de un objeto en el eje x, en el eje y o en ambos a la vez. Como parámetro se pasa los grados de deformación que queremos aplicar a un objeto determinado.

- skewX(gradoX)
- skewY(gradoY)
- skewXY(gradoX,gradoY)

Se adjunta ejemplo con las siguientes transformaciones:



Las transformaciones en 3D llevan una complejidad mayor que las 2D y no serán necesario utilizarlas en nuestra tare. Para aquel alumno que quiera profundizar se dejan las propiedades que intervienen en transformaciones 3D.

Hasta ahora hemos visto cómo transformar un objeto en 2D, pero CSS también nos permite realizar transformaciones en 3D, así tendremos además de los ejes X e Y el eje Z, con lo cual dispondremos de las propiedades translatez, scalez, rotatez y skewZ, para poder actuar a la vez en los tres ejes, tendremos las propiedades:

- translate3d
- scale3d
- rotate3d
- skew3d
- perspective(n). Nos permite indicar la distancia que existirá entre el observador y el elemento al cual vamos aplicar la transformación.
- perspective-origin: Nos permite definir el origen del observador indicando su posición en el eje X e Y. Por defecto sus valores son 50% 50%, es decir, están en el centro del elemento. Podemos indicar diferentes medidas como px, %, también podemos modificar la propiedad con los valores left, right y center para el eje X y top, bottom y center para el eje Y.
- transform-origin: Nos permite desplazar el eje del objeto sobre el que estamos aplicando la transformación.
- transforn-style: permite los valores preserve-3d y flat. El primero hará que si el objeto contiene objetos hijos se mantenga la transformación 3d para ellos.
- backface: Permite ver el dorso de un objeto. Es una propiedad que permitirá true o false.

A continuación dejamos un enlace donde se puede ver como animar en 3D un objeto. Lo relacionado con la animaciones se trata en puntos siguientes.

Animación 3d

2. TRANSICIONES CON CSS.

Una transición nos permitirá que un objeto o una propiedad experimente un cambio, donde nosotros podemos indicar como se realiza ese cambio y el tiempo que deseamos que dure dicho cambio. Así tendremos que definir un estado inicial y un estado final, el cómo se realiza este cambio y el tiempo que durará.

Para aplicar una transición a una, varias o todas las propiedades de un elemento tendremos que utilizar la propiedad transition. Esta propiedad controlará entre otras cuestiones el tiempo en el que se realiza el cambio, el ritmo que tendrá el cambio y el tiempo de retraso hasta que inicie el cambio.

La transición no se puede aplicar sobre todas las propiedades de un objeto, en el siguiente enlace podemos consultar las propiedades sobre las cuales podemos aplicar nuestra transición. Propiedades que pueden utilizarse en una transición.

Propiedades relacionadas con las transiciones

- transition: podemos indicar propiedades, tiempo en el que se produce la transición, ritmo, tiempo de retraso. Esto podemos realizarlo todo en una propiedad o en diferentes subpropiedades que a continuación citamos.

  Ejemplo: transition: all 3s ease 1s.

  Realizamos la transición sobre todas las propiedades del objeto, la transición durará 3 segundos, el ritmo será ease y la animación empezará con un segundo de retraso.

  Ejemplo: transition: width 3s linear, background-color 2s ease.

  Realizamos una transición width que durará 3 segundos con un ritmo linear y realizamos una transición sobre el color que durará 2 segundo con el rimto ease.

Como se ha visto en los ejemplos anteriores todos los parámetros de la transición se ha aplicado en la propiedad transition, pero existen otras subrpropiedades donde podemos configurar la transición. Así tendremos:

- transition-property: Indicamos las propiedades sobre las que queremos realizar la transición.
- transition-duration: Indicamos el tiempo en segundos que aplicaremos a cada transición.
- transition-timing-function: Indicamos el ritmo. En el siguiente enlace podemos ver los valores que puede tomar esta propiedad. Valores
- transition-delay: Indicamos el retraso con el que empezará la transición

En el siguiente ejemplo definimos la transición sobre dos propiedades, indicando los diferentes parámetros para cada propiedad.

Ejemplo:

```
transition-property: width, background-color;
transition-duration: 5s, 2s;
transition-timing-function: ease, linear;
transition-delay: 1s
```

En el siguiente ejemplo definimos la transición sobre dos propiedades, indicando para las mismas el mismo tiempo de duración y el mismo ritmo.

Ejemplo:

```
transition-property: width, background-color;
transition-duration: 5s;
transition-timing-function: linear;
transition-delay: 1s
```

En fichero adjunto, se facilitan diferentes transiciones.

La transformación se realizará al hacer clic sobre el objeto, pseudo-clase :active

La transformaciones se realizarán al posicionar el cursor sobre el objeto, pseudo-clase :hover

ease

ease-in

ease-out

linear-in-out

linear

3. ANIMACIONES @keyframe.

Como hemos vistos las transiciones se producen definiendo un estado inicial y un estado final, pero si quisiéramos definir estados intermedios, no podríamos, al menos con las herramientas vistas hasta el momento, para solucionar esto CSS nos facilita el uso de las animaciones (regla @keyframe).

Al igual que en la transición, en la animación podremos jugar con varias propiedades. Es decir una animación puede actuar sobre varias propiedades y sobre un objeto se pueden aplicar diferentes animaciones como veremos más adelante.

Para definir la animación, nos valdremos de lo que denominaremos como fotogramas clave, donde definiremos las propiedades de nuestro elemento. Podemos definir tantos fotogramas clave como queramos.

La animación más sencilla será la que definamos el estado inicial y final, en cada uno de estos estados (fotogramas clave) indicamos las propiedades sobre las que vamos a actuar y su valor.

```
@keyframes ejemplo_simple {
            from {
                background: gray;
                width: 100px;
                height: 100px;
                border: solid 2px black;
                margin: 100px;
            }
            to {
                background-color: olivedrab;
                width: 200px;
                height: 200px;
                transform: rotate(90deg);
                opacity: 0.5;
            }
        }
```

Una vez que hemos definido nuestra animación tendremos que definir cómo se va a producir dicho cambio sobre nuestro elemento, así nos valdremos de las siguientes subpropiedades:

- animation-name : Nombre de la animación
- animation-duration: Indicaremos en segundos (s) el tiempo que durará la animación
- animation-timing-function: Indicaremos al igual con las transiciones cómo se comportará nuestra animación.
- animation-iteration-count: Indicará el número de repeticiones que tendrá nuestra animación, le asignaremos un valor entero. Nuestra animación puede estar de forma ininterrumpidamente repitiéndose, para eso asignaremos el valor infinite.

- animation-direction: Puede realizarse de la forma en que hemos definido la animación o al revés, es decir, empezaremos por el estado final y terminado con el estado inicial, para eso asignaremos los valores alternate o reverse.
- animation-delay: Tiempo de retraso con el que empezará la animación.

Estas son las principales propiedades que utilizaremos para definer nuestra animación, dejamos otras para profundizar como son animation-play-state y animation-fill-mode

En el anterior ejemplo, tenemos un ejemplo inicial y final, lo cual puede parecerse a la transición, pero con la regla @keyframe, podemos definir estados intermedios o fotogramas clave, tantos como queramos. De esta forma vamos configurando nuestra animación, indicando el comportamiento de nuestro elemento en cada fotograma clave (o porcentaje). En cada porcentaje tendremos indicar (como se ven en el siguiente ejemplo) .las propiedades con los respectivos valores que van a tener.

En el siguiente ejemplo definimos tres estados,

```
@keyframes simulaciona {
            0% {
                top: 0px;
                right: 0px;
                background-color: white;
            }
            50% {
                top: 250px;
                right: 125;
                opacity: 0.8;
                background: blue;
            }
            100% {
                top: 150px;
                right: 0;
                opacity: 0.2;
                background: red;
            }
        }
```

Una vez definida nuestra animación indicamos como se va a aplicar esta, para eso definimos las propiedades citadas anteriormente.

Así sobre el objeto animaciona le aplicaremos la animacion simulaciona, que durará 5 segundos y se repetira de forma infinita.

```
    #animaciona {
            animation-name: simulaciona;
            animation-duration: 5s;
```

```
            animation-iteration-count: infinite;
        }
```

Se deja un fichero con un ejemplo de animación.

Ejemplo de animación con from - to

Ejemplo de animación definiendo dos fotogramas clave 0% y 100%