CSS ANIMATIONS

CSS permite animar elemento HTML sem usar JavaScript ou flash, um elemento é animado quando ele gradualmente vai de um estilo para outro, é possível alterar muitas propriedades CSS de muitas maneiras. Para usar animações CSS é preciso de alguns keyframes específicos. Os keyframes contém os estilos que um elemento vai possuir em um certo momento.

@KEYFRAME

Quando um estilo é especificado dentro de uma regra @keyframe, a animação vai mudar gradualmente para do estilo atual para o novo estilo em um determinado momento. Para fazer a animação funcionar é preciso vincular a animação a um elemento. No seguinte exemplo a animação “example” é vinculada ao elemento através da propriedade **animation-name**. A animação vai terminar após 4 segundos, e vai gradualmente alterar a propriedade background-color do elemento de “red” para “yellow”.

```
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}
```

A propriedade **animation-duration** define **quanto tempo vai durar a animação** até que ela esteja completa. Se essa propriedade não for especificada, nenhuma animação vai acontecer, já que o valor padrão é 0s. Para representar os estágios da animação é possível utilizar “from” e “to” dentro do keyframes. Porém o uso de valores em porcentagem permite adicionar mais estágios a animação. No seguinte exemplo o background-color vai ser alterado do elemento quando a animação estiver 25% completa, 50% completa, e novamente quando a animação for 100% completa.

```
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}

@keyframes example {
  0%   {background-color: red;}
  25%  {background-color: yellow;}
  50%  {background-color: blue;}
  100% {background-color: green;}
}
```

animation-delay

A propriedade **animation-delay** especifica o **tempo de atraso para a animação ser iniciada**. O seguinte exemplo a animação vai acontecer após 2 segundos de delay.

```
div {
  width: 100px;
  height: 100px;
  position: relative;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
  animation-delay: 2s;
}
```

É possível ainda acrescentar valores negativos, esses valores fazem a animação iniciar como se já estivesse sendo reproduzida por *N* segundos.

animation-interation-count

A propriedade **animation-interation-count** especifica o **número de vezes em que a animação vai ser executada**. No seguinte exemplo a animação vai ser executada 3 vezes, e após isso parar. No outro exemplo é utilizado o **valor "infinite"** para fazer a animação continuar **para sempre**:

```
div {
  width: 100px;
  height: 100px;
  position: relative;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
  animation-iteration-count: 3;
}
```

```
div {
  width: 100px;
  height: 100px;
  position: relative;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
  animation-iteration-count: infinite;
}
```

animation-direction

A propriedade **animation-direction** especifica se a animação vai ser executada **de forma progressiva** (para frente), **regressiva** (para trás) ou em **ciclos alternados**. Para o valor “alternate” funcionar é preciso que a propriedade animation-interation-count esteja definida com um valor maior ou igual a dois. O exemplo a seguir usa o valor "alternate" fazendo com que a animação seja executada e depois executada novamente de forma inversa.

```
div {
  width: 100px;
  height: 100px;
  position: relative;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
  animation-iteration-count: 2;
  animation-direction: alternate;
}
```

A propriedade animation-direction pode ter os seguintes valores:

- **normal**
- **reverse**
- **alternate**
- **alternate-reverse**

animation-fill-mode

A propriedade **animation-fill-mode especifica o estilo do elemento quando a animação não está sendo executada** (antes de começar, após ela terminar e ambos). Possuí os seguintes valores:

- **none** - **default**. A animação não aplicará nenhum estilo ao elemento antes ou depois de ser executado
- **forwards** - O elemento manterá os valores de estilo definidos pelo **último keyframe** (dependendo das propriedades animation-direction e da animation-interation-count)
- **backwards** - O elemento recebera os valores de estilo definidos pelo **primeiro keyframe** (dependendo da animation-direction) manterá esse estilo durante o tempo de delay.
- **both**- A animação fará as duas regras anteriores.

Sem o uso do animation-fill-mode o elemento vai começar e terminar a animação de maneira brusca, com os estilos padrões sendo atribuídos ao elemento. No seguinte exemplo a animação vai manter a última alteração de estilo definida no keyframe.

```
div {
  width: 100px;
  height: 100px;
  background: red;
  position: relative;
  animation-name: example;
  animation-duration: 3s;
  animation-delay: 2s;
  animation-fill-mode: backwards;
}
```

animation-timing-funcion

A propriedade **animation-fill-mode** especifica a **curva de velocidade da animação**. Possuí os seguintes valores:

- **ease** – default. Especifica uma animação que vai começar devagar, ficar rápida, e terminar devagar.
- **linear** – Especifica uma animação com a mesma velocidade do começo até o fim.
- **ease-in** – Especifica uma animação com uma velocidade devagar no começo.
- **ease-out** - Especifica uma animação com uma velocidade devagar no final.
- **ease-in-out** - Especifica uma animação com uma velocidade devagar no começo e no final.
- **cubic-bezier(n,n,n,n)** – Permite especificar os valores da velocidade em uma função.

animation shorthand

A propriedade **animation** é uma **maneira encurtada de usar todas as propriedades anteriores** em uma única linha de código. No seguinte exemplo utiliza 6 propriedades de animação diferentes.

```
div {
  animation-name: example;
  animation-duration: 5s;
  animation-timing-function: linear;
  animation-delay: 2s;
  animation-iteration-count: infinite;
  animation-direction: alternate;
}
```

O mesmo efeito de animação acima pode ser obtido usando a propriedade de animation abreviada.

```
div {
  animation: example 5s linear 2s infinite alternate;
}
```

```
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  animation: myfirst 5s linear 2s infinite alternate;
}

@keyframes myfirst {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}
```

CSS TRANSITIONS

As transições CSS permitem alterar os valores de propriedade de forma suave durante um determinado período de duração. Se o tempo de duração não for especificado a transição não vai ter nenhum efeito pois o valor default é 0. No seguinte exemplo um elemento com dimensões de 100px x 100px recebe uma alteração no valor da propriedade width quando a pseudoclasse (“hover”) do elemento é ativada. A transição de desse valor é feita de forma gradual e suave devido a propriedade transition.

```
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s;
}
div:hover {
  width: 300px;
}
```

transition-delay

A propriedade **transition-delay** especifica um atraso (em segundos) no efeito de transição.

```
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 3s;
  transition-delay: 1s;
}

div:hover {
  width: 300px;
}
```

transition-timing-function

A propriedade **transition-timing-function** especifica a curva de velocidade em que a transição vai acontecer, assim como nas animações.

- **ease** – default. Especifica uma transição que vai começar devagar, ficar rápida, e terminar devagar.
- **linear** – Especifica uma transição com a mesma velocidade do começo até o fim.
- **ease-in** – Especifica uma transição com uma velocidade devagar no começo.
- **ease-out** - Especifica uma transição com uma velocidade devagar no final.
- **ease-in-out** - Especifica uma transição com uma velocidade devagar no começo e no final.
- **cubic-bezier(n,n,n,n)** – Permite especificar os valores da velocidade em uma função.

transition shorthand

No seguinte exemplo a transição de estilo foi especificada em 4 diferentes propriedades:

```
div {
  transition-property: width;
  transition-duration: 2s;
  transition-timing-function: linear;
  transition-delay: 1s;
}
```

É possível simplificar e determinar todas essas propriedades em apenas uma:

```
div {
  transition: width 2s linear 1s;
}
```

(As propriedades devem ser definidas nessa sequência: transition-property, transition-duration, transition-timing-function e transition-delay)

É possível ainda especificar a transição de alteração de diversas propriedades em tempos diferentes, adicionando o sinal gráfico de vírgula “,” após especificada o nome cada propriedade e seus devidos tempos de duração, como no seguinte exemplo.

```
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s, height 2s, transform 2s;
}

div:hover {
  width: 300px;
  height: 300px;
  transform: rotate(180deg);
}
```

```
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s linear 1s, height 4s ease 2s;
}

div:hover {
  width: 300px;
  height: 300px;
}
```