Animation in Css

CSS @keyframes are used to define animations by specifying keyframes (intermediate steps) along the animation timeline.

Syntax:

```
@keyframes animation-name {
    0% {
      /* starting styles */
    }
    50% {
      /* middle styles */
    }
    100% {
      /* ending styles */
    }
  }
```

There are 2ways to write animate

1. using from { } or 0%
    - from or 0% = starting state
2. using to{} or 100%
    - to or 100% = ending state

There are also some other properties that will make your animation smooth and ease in viewing

Some are listed below:

- animation-name: Name of the keyframes.
- animation-duration: How long the animation takes to complete one cycle.
- animation-timing-function: How the animation progresses over time (e.g. ease, linear, ease-in-out).
- animation-delay: Wait time before the animation starts.

- animation-iteration-count: How many times it repeats (infinite for forever). or 4s,3s etc.
- animation-direction: Whether it runs forward, backward, or alternates.
- animation-fill-mode: Whether styles persist after animation (forwards, backwards, both, or none).

Example 1

```
in html
<div class="box"> </div>

in css
<style>

.box {
    width: 350px;
    height: 150px;
    position: relative;

     /* short hand property */
    animation: demo 4s ease-in-out 1s 3 alternate forwards;
     /* short hand property */

    /* long hand property */
    /* animation-duration: 4s;
    animation-timing-function: ease-in-out;
    animation-delay: 1s;
    animation-iteration-count: 3;
    animation-direction: alternate;
    animation-fill-mode: forwards; */
    /* long hand property */

    }

@keyframes demo {
    from {
        left: 0px;
```

```
            background-image: url(https://cf-img-a-in.tosshub.com/sites/visualstory/wp/2024/07/opener-w-Bugatti-3.webp?size=*:675);
            background-size: cover;
        }

        to{
            left: 700px;
            background-image: url(https://cf-img-a-in.tosshub.com/sites/visualstory/wp/2024/07/opener-w-Bugatti-3.webp?size=*:675);
            background-image: url(https://cf-img-a-in.tosshub.com/sites/visualstory/wp/2024/07/opener-w-Bugatti-3.webp?size=*:675);
            background-size: cover;

        }
    }

    <style>
```

Example 2

```
        in html
        <div class="box">
            <span> </span>
            <h2>Animated Border</h2>
        </div>

        in css
        <style>
        .box {
            position: relative;
            height: 250px;
            width: 250px;
            background: rgba(0, 0, 0, 0.8);
            border-radius: 10px;
            display: flex;
            justify-content: center;
            align-items: center;
```

```
  overflow: hidden;
}

.box::before,.box::after {
  content: "";
  position: absolute;
  height: 150%;
  width: 150%;
  background-image: linear-gradient(#e9db1d,#000);
  animation: animate 5s linear infinite;
  animation-delay: -1.5s;
}

@keyframes animate {
  0% {
    transform: rotate(0deg);
  }

  100% {
    transform: rotate(360deg);
  }
}

.box span {
  position: absolute;
  inset: 9px;
  border-radius: 10px;
  background: #0c1022;
  z-index: 1;
}

.box h2 {
  position: relative;
  z-index: 2;
  color: #fff;
```

```
    font-size: 1.5rem;
    text-align: center;
    padding: 0 20px;
  }
<style>
```

Example 3

```
in html
<ul class="animated-text">
    <li>q</li>
    <li>u</li>
    <li>i</li>
    <li>c</li>
    <li>k</li>
    <li>X</li>
    <li>p</li>
    <li>e</li>
    <li>r</li>
    <li>t</li>
</ul>

in Css
<style>
ul.animated-text {
    list-style-type: none;
    padding: 0;
    margin: 0;
  }

  .animated-text li {
    display: inline-block;
    font-size: 2rem;
    margin: 10px;
```

```
      transform: scale(1);
      animation: none;
    }

    @keyframes scaleUpDown {
      0%   { transform: scale(1); }
      5%   { transform: scale(1.5); }
      10%  { transform: scale(1); }
      100% { transform: scale(1); }
    }

    .animated-text li:nth-child(1)  { animation: scaleUpDown 10s linear 0s 1 forwards; }
    .animated-text li:nth-child(2)  { animation: scaleUpDown 10s linear 1s 1 forwards; }
    .animated-text li:nth-child(3)  { animation: scaleUpDown 10s linear 2s 1 forwards; }
    .animated-text li:nth-child(4)  { animation: scaleUpDown 10s linear 3s 1 forwards; }
    .animated-text li:nth-child(5)  { animation: scaleUpDown 10s linear 4s 1 forwards; }
    .animated-text li:nth-child(6)  { animation: scaleUpDown 10s linear 5s 1 forwards; }
    .animated-text li:nth-child(7)  { animation: scaleUpDown 10s linear 6s 1 forwards; }
    .animated-text li:nth-child(8)  { animation: scaleUpDown 10s linear 7s 1 forwards; }
    .animated-text li:nth-child(9)  { animation: scaleUpDown 10s linear 8s 1 forwards; }
    .animated-text li:nth-child(10) { animation: scaleUpDown 10s linear 9s 1 forwards; }
  <style>
```

The transform property in CSS allows you to modify the position, size, and shape of an element.
You can use it to apply various transformations such as

- translating
- rotating
- scaling
- skewing

2D Transform

- 2D Transforms involve transformations on the X and Y axes
    1. Translate Moves from X and Y axes

Syntax

```
    translate(x, y) – Moves from X and Y axes.
    selector {
    transform: translate(50px, 100px);
    }


 in html
    <div class="box">
        <h1>hover me</h1>
    </div>

 in Css
 <style>
 .box{
    width: 400px;
    height: 100px;
    background-color: red;
   }

   .box:hover{
    transform:translate(50px, 100px);
    transition: 1s ease-in-out; /* transition for smooth effect */
    cursor: pointer; /* hand gesture symbol*/
   }

 </style>
```

2. Rotate : Rotates the element around from X axis Y axis

Syntax

```
    rotate(x, y) – Rotates the element around a 2D point
    selector {
```

```
    transform: rotateX(35deg) rotateY(50deg)
    }


in html
    <div class="box">
        <h1>hover me</h1>
    </div>

in Css
<style>
.box{
    width: 400px;
    height: 100px;
    background-color: red;
  }

  .box:hover{
    transform: rotateX(35deg) rotateY(50deg);
    transition: 1s ease;
    /*can apple +ve and -ve value both*/
  }
</style>
```

3. Scale : Scales the element by x and y along the horizontal and vertical axes.

Syntax

```
    scale(x, y) - Scales the element by X axis and Y axis .
    selector {
        transform: scale(1.5, 2);
      }


in html
    <div class="box">
```

```
        <h1>hover me</h1>
    </div>

in Css
<style>
.box{
    width: 400px;
    height: 100px;
    background-color: red;
  }

  .box:hover{
     transform: scale(.9,.6);
     transition: 1s ease;

     /*(point values for zoom in)
     normal values for zoom out
     does not accept px units*/
  }
</style>
```

4. Skews : Skews the element by x and y along the horizontal and vertical axes.

Syntax

```
    skew(x, y) - Skews the element along X and Y axes .
    selector {
        transform: skew(30deg, 20deg);
      }


in html
    <div class="box">
        <h1>hover me</h1>
    </div>
```

```
in Css
<style>
.box{
    width: 400px;
    height: 100px;
    background-color: red;
  }

  .box:hover{
    transform: skew(10deg, 20deg);
    transition: 2s ease;
/* it accept degree unit */
  }
</style>
```

3D Transform

3D transforms involve transformations along X, Y, and Z axes.
3D transforms also include perspective, which creates depth and makes the transformations looks realistic.

1. Translate Moves from X , Y & Z axis

Syntax

```
translate3d(x, y, z) – Moves the element along X, Y, and Z axes.
selector {
  transform: translate3d(50px, 100px, 200px);
}
```

```
in html
<div class="box">
<h1>hover me</h1>
</div>

in Css
```

```
<style>
.box{
width: 400px;
height: 100px;
background-color: red;
}

.box:hover{
    transform: translate3d(50px, 100px, 700px);
    transition: 1s ease-in-out;
}

</style>
```

2. Rotate : Rotates the element around from X , Y & Z axis

Syntax

```
    rotateX(deg) - Rotates the element around the X axis.
    rotateY(deg) - Rotates the element around the Y axis.
    rotateZ(deg) - Rotates the element around the Z axis it is similar to the 2D rotate().
    selector {
      transform: rotateX(45deg) rotateY(30deg) rotateZ(10deg);
    }


in html
<div class="box">
<h1>hover me</h1>
</div>

in Css
<style>
.box{
width: 400px;
height: 100px;
background-color: red;
}

.box:hover{
    transform: rotateX(45deg) rotateY(45deg) rotateZ(45deg);
    transition: 5s ease;
    box-shadow: 10px 10px 10px red;
}
</style>
```

3. Scale : scale3d(x, y, z) – Scales the element along X, Y, and Z axes.

Syntax

```
    scale3d(x, y, z) - Scales the element along X, Y, and Z axes.
    selector {
      transform: scale3d(1.5, 1.5, 2);
    }


in html
<div class="box">
<h1>hover me</h1>
</div>

in Css
<style>
.box{
width: 400px;
height: 100px;
background-color: red;
}

.box:hover{

transform: scale3d(.8, .4, .1);
transition: 1s ease;
}
</style>
```

4. 3D skew effect, but it’s implemented differently compared to 2D skew.
   While 2D skewing happens along the X and Y axes, 3D skewing
   involves skewing along the X, Y, and Z axes.
   However, there's no direct skew3d() function in CSS.
   Instead, you can achieve a 3D skew effect using matrix3d()

   The matrix3d() function is used for applying a 3D transformation to an element.
   It allows you to define the transformation using a 4x4 matrix,
   which consists of 16 values.

Syntax

```
    matrix3d(a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p)
```

```
in html
<div class="box">
<h1>hover me</h1>
</div>

in Css
<style>
.box{
width: 400px;
height: 100px;
background-color: red;
}

.box:hover{
    transform: matrix3d(1,.5, 0, 0, .7, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
    transition: 1s ease;
}
</style>
```

The perspective property in CSS is used to create a 3D space for elements, giving the illusion of depth.

syntax

```
    perspective:<length>
```

The distance from the viewer to the element.
It can be a value in pixels (px), ems (em), rems (rem), percentages (%), etc.
Smaller values create a more intense 3D effect.

Example

```
in html
<div class="container">
    <div class="box">
        <h1>QuickXpert</h1>
        <img src="https://cdn.pixabay.com/photo/2016/12/14/13/59/ball-1906468_1280.png" alt="demo">
    </div>
</div>

in Css
<style>

.container {
width: 500px;
height: 500px;
/* as u reduce the px it will go deeper inside */
perspective: 1000px;
margin: 10rem auto;
}

h1{
```

```
text-shadow: 10px 10px 10px red;
background: transparent;
}

.box {
width: 300px;
height: 100px;
transform: rotateY(45deg) rotateX(45deg) rotateZ(45deg);
transition:1s;
cursor: pointer;
text-align: center;
background: transparent;
}
.box:hover {
transform: rotateX(0deg) rotateY(0deg) rotateZ(0deg);
}

img{
width: 300px;
height: 300px;
}
</style>
```