Installation and Usage

Installing

Install with npm:

```
$ npm install animate.css --save
```

Or install with Yarn (this will only work with appropriate tooling like Webpack, Parcel, etc. If you are not using any tool for packing or bundling your code, you can simply use the CDN method below):

```
$ yarn add animate.css
```

Import it into your file:

```
import 'animate.css';
```

Or add it directly to your webpage using a CDN:

```
<head>
  <link
    rel="stylesheet"
    href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/4.1.1/animate.min.css"
  />
</head>
```

Basic usage

After installing Animate.css, add the class `animate__animated` to an element, along with any of the animation names (don't forget the `animate__` prefix!):

```
<h1 class="animate__animated animate__bounce">An animated element</h1>
```

That's it! You've got a CSS animated element. Super!

> Animations can improve the UX of an interface, but keep in mind that they can also get in the way of your users! Please read the best practices and gotchas sections to bring your web-things to life in the best way possible.

Using `@keyframes` Even though the library provides you a few helper classes like the `animated` class to get you up running quickly, you can directly use the provided animations `keyframes`. This provides a flexible way to use Animate.css with your current projects without having to refactor your HTML code.

Example:

```
.my-element {
  display: inline-block;
  margin: 0 0.5rem;

  animation: bounce; /* referring directly to the animation's @keyframe declaration */
```

```
  animation-duration: 2s; /* don't forget to set a duration! */
}
```

Be aware that some animations are dependent on the `animation-timing` property set on the animation's class. Changing or not declaring it might lead to unexpected results.

CSS Custom Properties (CSS Variables) Since version 4, Animate.css uses custom properties (also known as CSS variables) to define the animation's duration, delay, and iterations. This makes Animate.css very flexible and customizable. Need to change an animation duration? Just set a new value globally or locally.

Example:

```
/* This only changes this particular animation duration */
.animate__animated.animate__bounce {
  --animate-duration: 2s;
}

/* This changes all the animations globally */
:root {
  --animate-duration: 800ms;
  --animate-delay: 0.9s;
}
```

Custom properties also make it easy to change all your animation's time-constrained properties on the fly. It means that you can have a slow-motion or time-lapse effect with a javascript one-liner:

```
// All animations will take twice the time to accomplish
document.documentElement.style.setProperty('--animate-duration', '2s');

// All animations will take half the time to accomplish
document.documentElement.style.setProperty('--animate-duration', '.5s');
```

Even though some aging browsers do not support custom properties, Animate.css provides a proper fallback, widening its support for any browser that supports CSS animations.