Usage with Javascript

You can do a whole bunch of other stuff with animate.css when you combine it with Javascript. A simple example:

```
const element = document.querySelector('.my-element');
element.classList.add('animate__animated', 'animate__bounceOutLeft');
```

You can detect when an animation ends:

```
const element = document.querySelector('.my-element');
element.classList.add('animate__animated', 'animate__bounceOutLeft');

element.addEventListener('animationend', () => {
  // do something
});
```

or change its duration:

```
const element = document.querySelector('.my-element');
element.style.setProperty('--animate-duration', '0.5s');
```

You can also use a simple function to add the animations classes and remove them automatically:

```
const animateCSS = (element, animation, prefix = 'animate__') =>
  // We create a Promise and return it
  new Promise((resolve, reject) => {
    const animationName = `${prefix}${animation}`;
    const node = document.querySelector(element);

    node.classList.add(`${prefix}animated`, animationName);

    // When the animation ends, we clean the classes and resolve the Promise
    function handleAnimationEnd(event) {
      event.stopPropagation();
      node.classList.remove(`${prefix}animated`, animationName);
      resolve('Animation ended');
    }

    node.addEventListener('animationend', handleAnimationEnd, {once: true});
  });
```

And use it like this:

```
animateCSS('.my-element', 'bounce');

// or
animateCSS('.my-element', 'bounce').then((message) => {
  // Do something after the animation
});
```

If you had a hard time understanding the previous function, have a look at const, classList, arrow functions, and Promises.