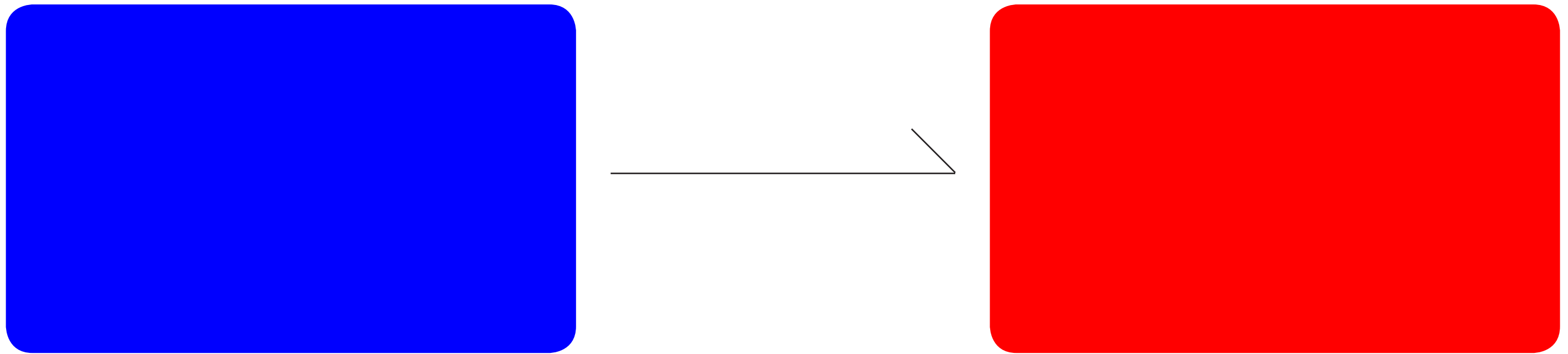


# Web Animations with CSS

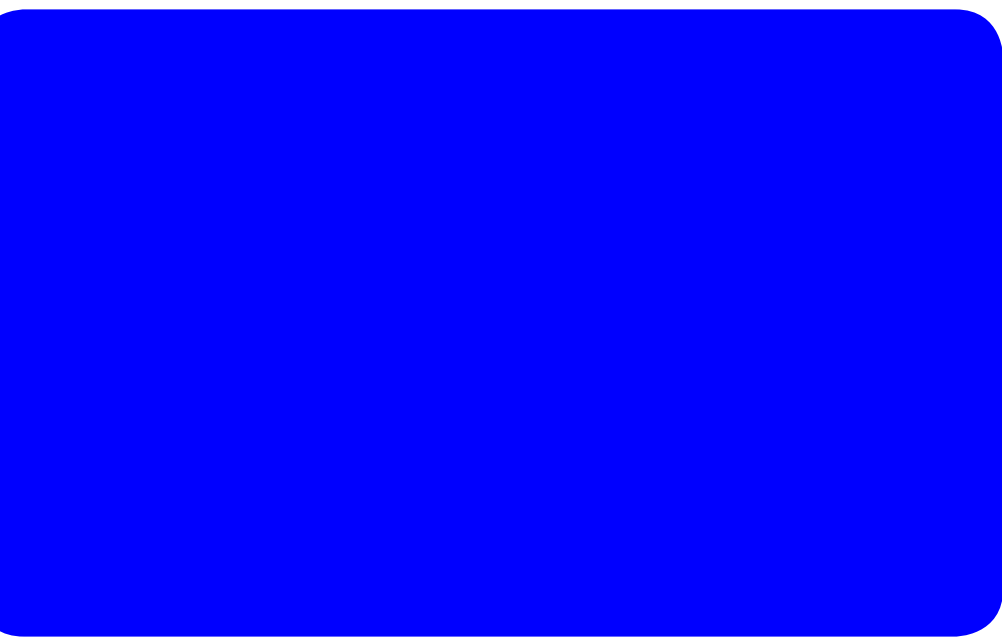
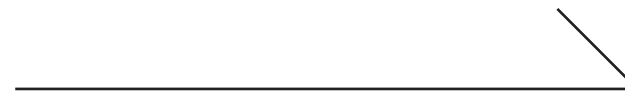
# What is a transition?

- Transitions cause changes to property and take place over a period of time.

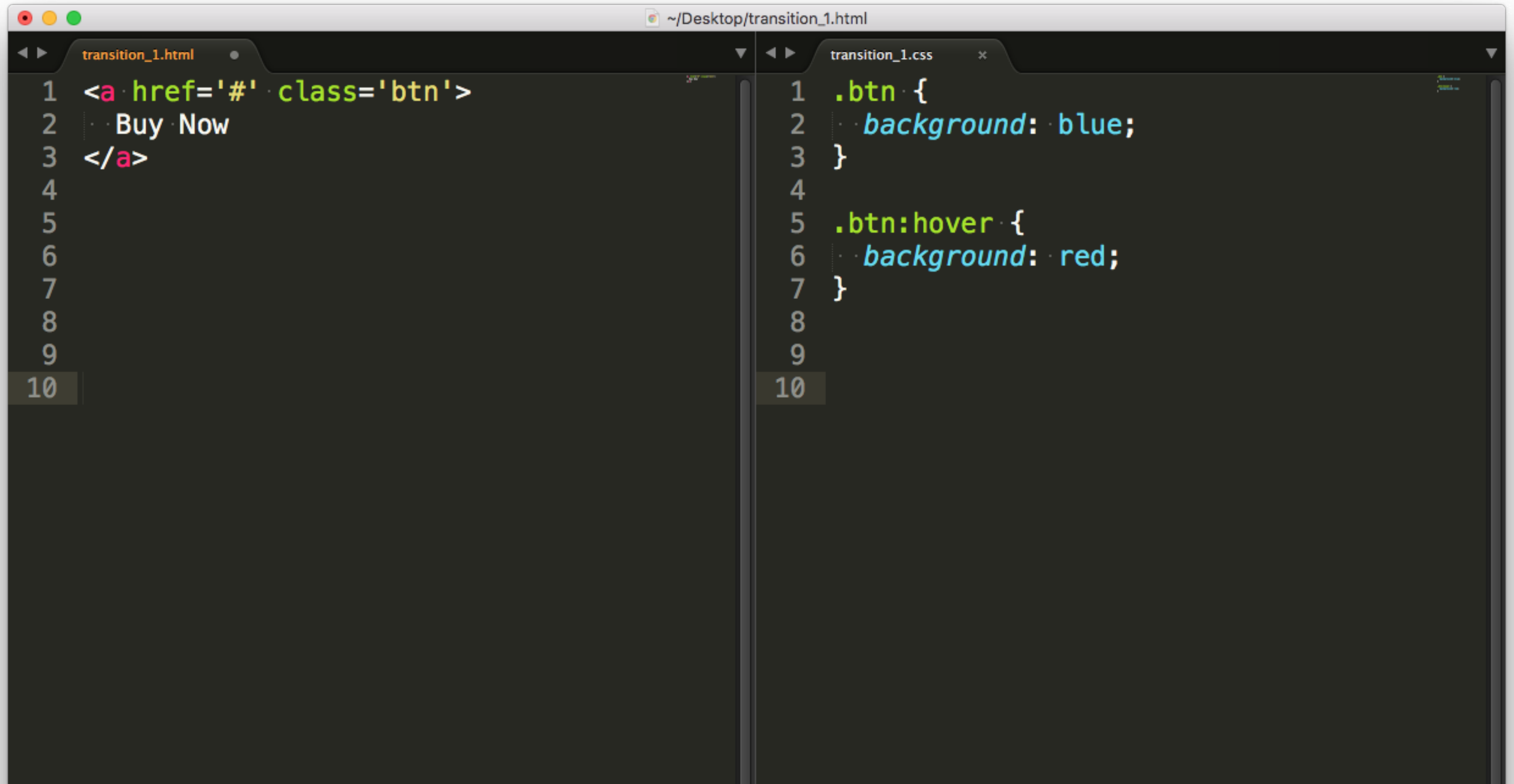


Hover State

Transitions allow the  
change to happen over  
a “duration” of time



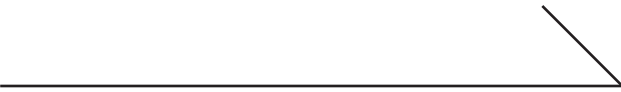
# First style start and end state



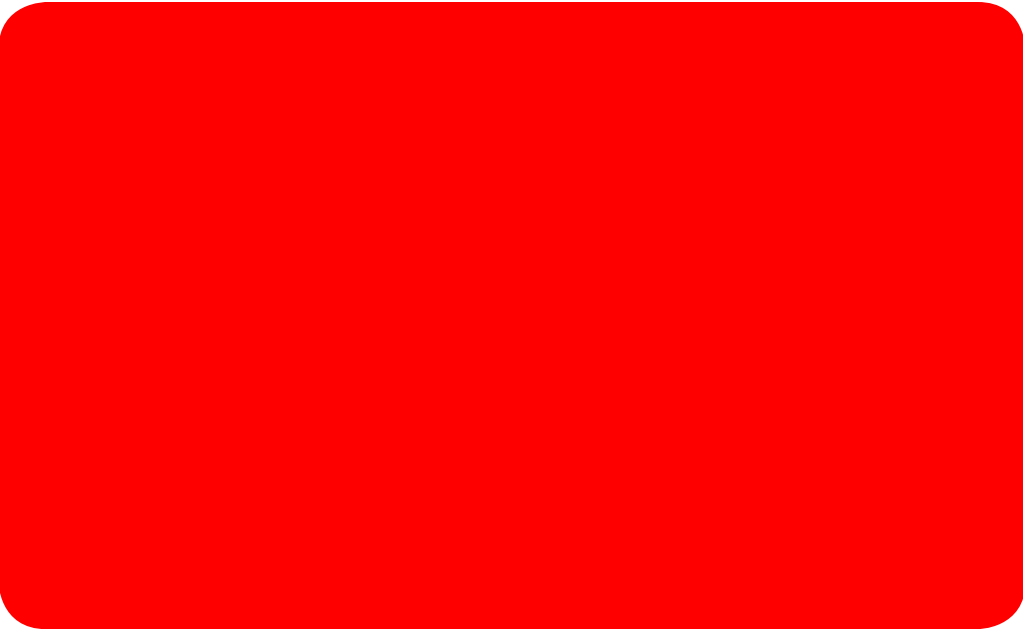
The image shows a code editor with two files open: `transition_1.html` and `transition_1.css`. The HTML file contains a single line of code for a button with the class `btn`. The CSS file defines two styles for the `.btn` class: a default blue background and a red background for the `:hover` state. The code is as follows:

```
1 <a href='#' class='btn'>  
2   Buy Now  
3 </a>  
4  
5  
6  
7  
8  
9  
10
```

```
1 .btn {  
2   background: blue;  
3 }  
4  
5 .btn:hover {  
6   background: red;  
7 }  
8  
9  
10
```



Instant

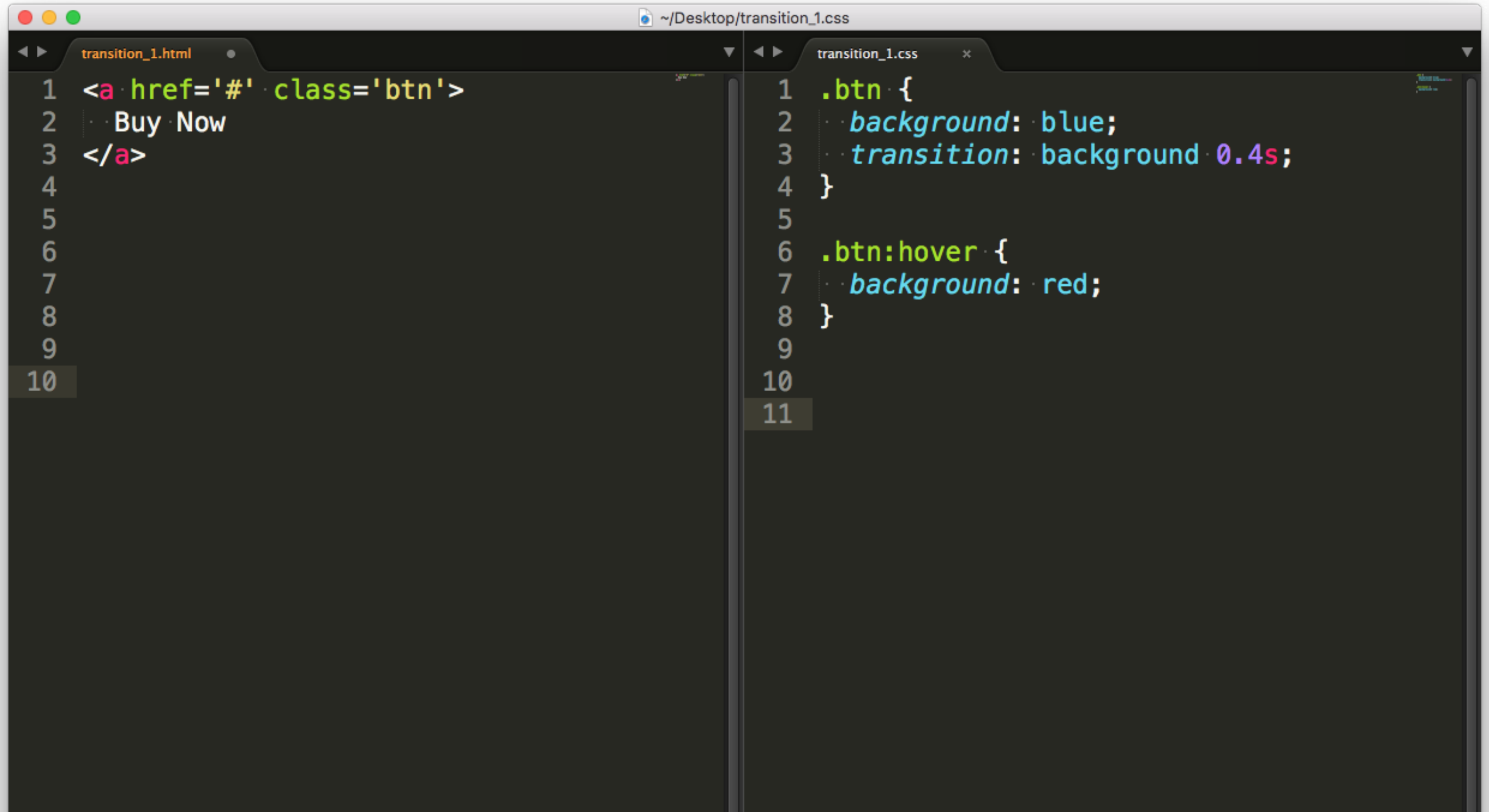


# Transition Recipe

It works like this:

```
transition: <property> <duration>;
```

# Transition Recipe



The image shows a code editor with two files open: `transition_1.html` and `transition_1.css`. The `transition_1.html` file contains an anchor tag with the text "Buy Now". The `transition_1.css` file contains CSS rules for the `.btn` class, including a background color of blue and a transition duration of 0.4s. A hover state is also defined, changing the background color to red.

```
1 <a href='#' class='btn'>
2   Buy Now
3 </a>
4
5
6
7
8
9
10
11
```

```
1 .btn {
2   background: blue;
3   transition: background 0.4s;
4 }
5
6 .btn:hover {
7   background: red;
8 }
9
10
11
```

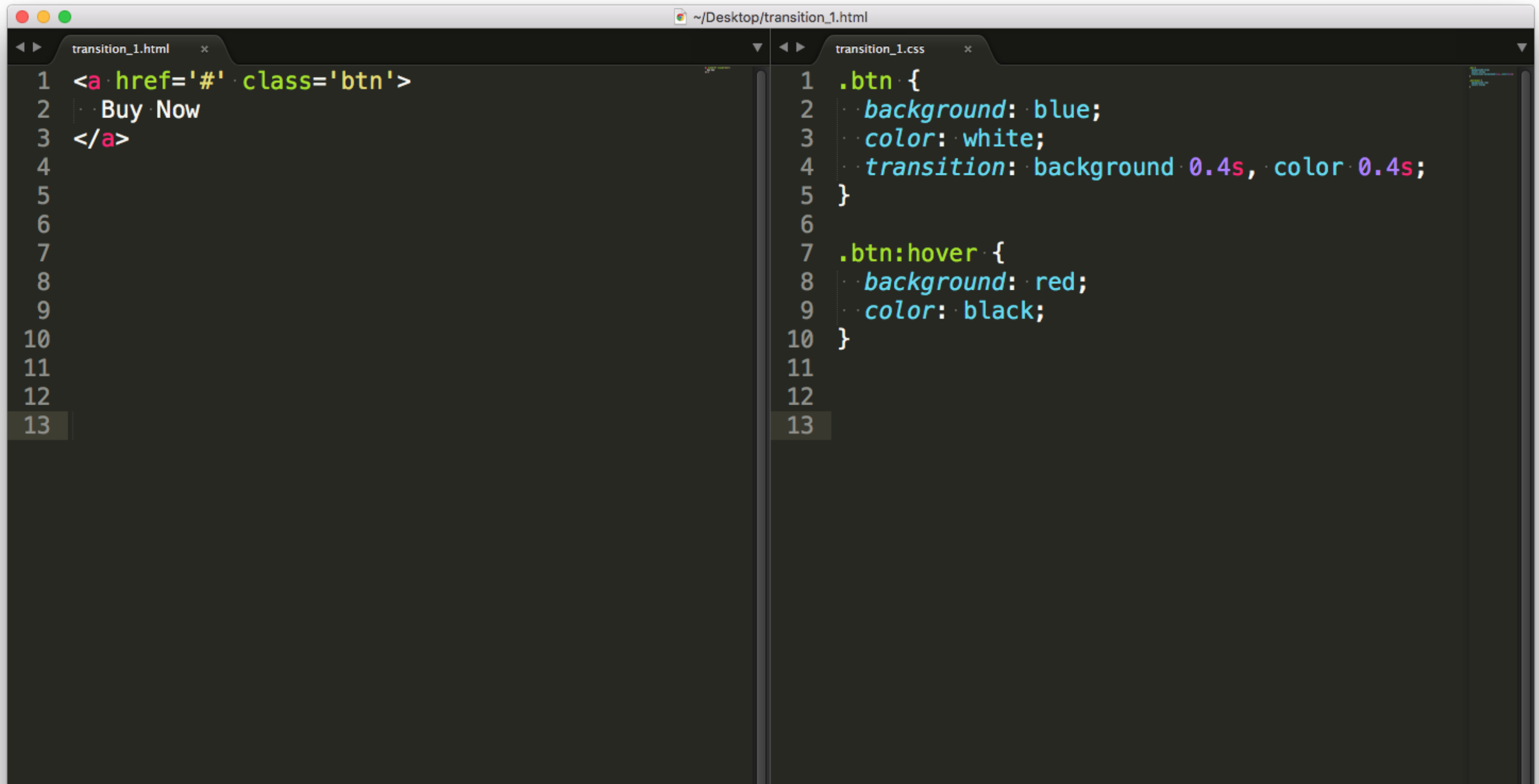
# Transition Multiple Properties

- You can transition multiple comma-separated properties

transition: <property> <duration>, <property> <duration>;



# Transition Multiple Properties

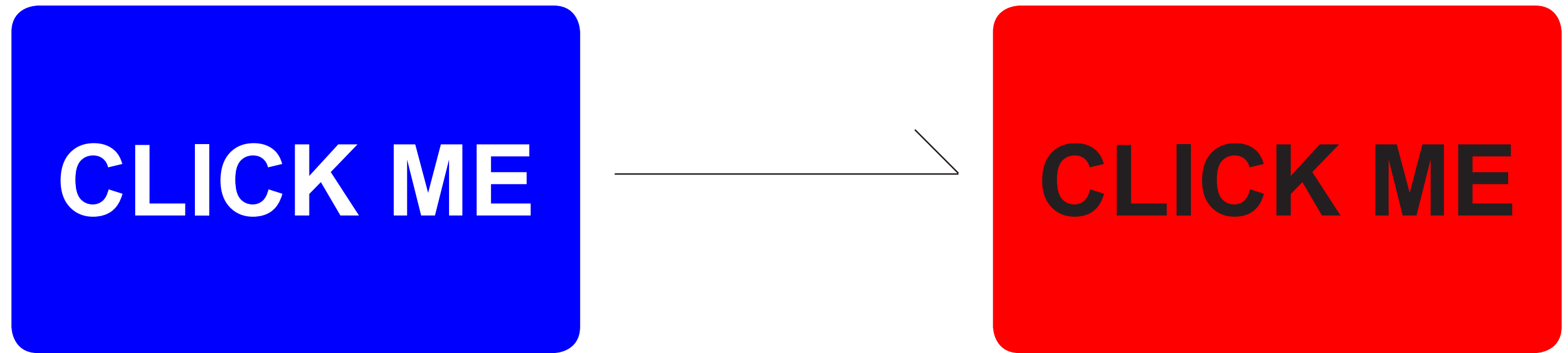


The image shows a code editor with two files: `transition_1.html` and `transition_1.css`. The HTML file contains a button element with the class `btn` and the text "Buy Now". The CSS file defines the default style for the `.btn` class (blue background, white color) and a `.btn:hover` state (red background, black color). A transition is applied to both the background and color properties, with a duration of 0.4 seconds for each.

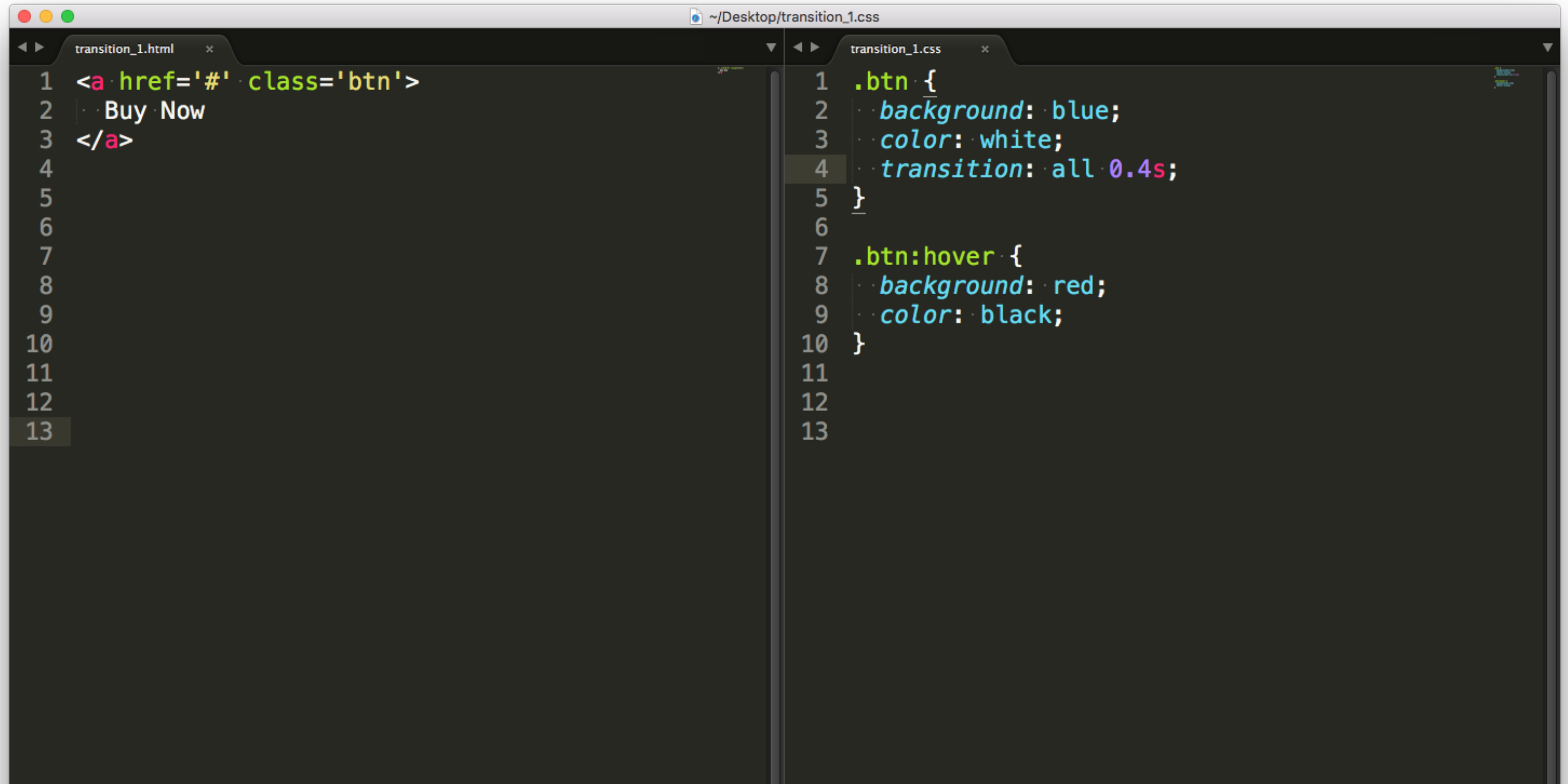
```
1 <a href='#' class='btn'>
2   Buy Now
3 </a>
4
5
6
7
8
9
10
11
12
13
```

```
1 .btn {
2   background: blue;
3   color: white;
4   transition: background 0.4s, color 0.4s;
5 }
6
7 .btn:hover {
8   background: red;
9   color: black;
10 }
11
12
13
```

# Transition Multiple Properties



# A Keyword to Transition All Properties



The image shows a code editor with two files open: `transition_1.html` and `transition_1.css`. The `transition_1.html` file contains an anchor tag with the text "Buy Now". The `transition_1.css` file contains CSS rules for the `.btn` class, including a default style with a blue background and white color, and a `:hover` style with a red background and black color. A transition of all properties over 0.4 seconds is applied to the `.btn` class.

```
1 <a href='#' class='btn'>
2   Buy Now
3 </a>
4
5
6
7
8
9
10
11
12
13
```

```
1 .btn {
2   background: blue;
3   color: white;
4   transition: all 0.4s;
5 }
6
7 .btn:hover {
8   background: red;
9   color: black;
10 }
11
12
13
```

# Full Transition Recipe

- Order is irrelevant as long as you have your duration number specified before the delay number.

```
.btn {
```

```
    transition: <property> <duration> <timing-function> <delay>;
```

```
}
```

# Defaults

- these properties are given defaults if you don't specify something for each of them.

```
.btn {
```

```
  transition: <property> <duration> <timing-function> <delay>;
```

```
}
```

Default: 'all'

Default: 'ease'

Default: 0

# When to use vendor prefixes

- All of these examples have been without vendor prefixes, but you might need to include them. They look like this:

```
.btn {  
  -webkit-transition: <property> <duration>;  
  -moz-transition: <property> <duration>;  
  -ms-transition: <property> <duration>;  
  transition: <property> <duration>;  
}
```

# When to use vendor prefixes

Use a site like [caniuse.com](http://caniuse.com) to check

# What can you use transition between?

Basically any css property that has a middle state.

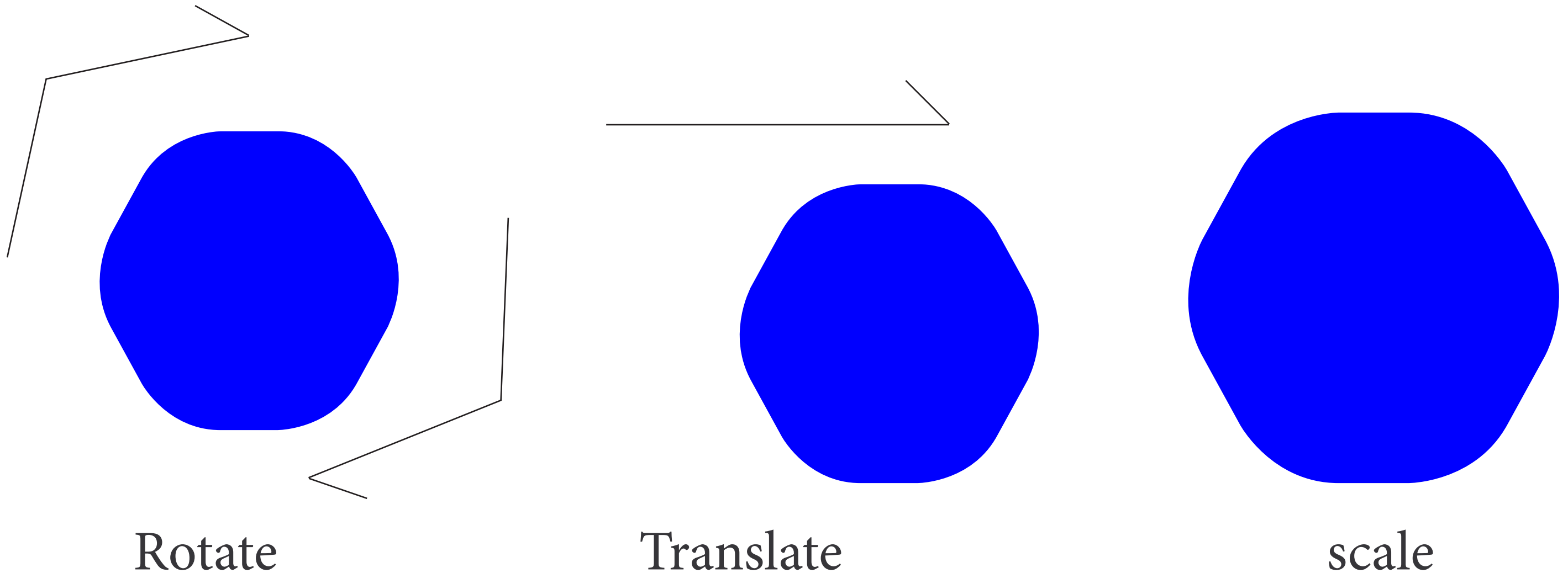
Easier to think about what you CAN'T transition between.

What about the 'display' property?



# What is a transform?

- CSS transformation let you modify elements in their coordinate space. They can be rotated, translated, scaled, and skewed



# Rotate recipe

- Rotate takes any value with a degree or turn suffix.

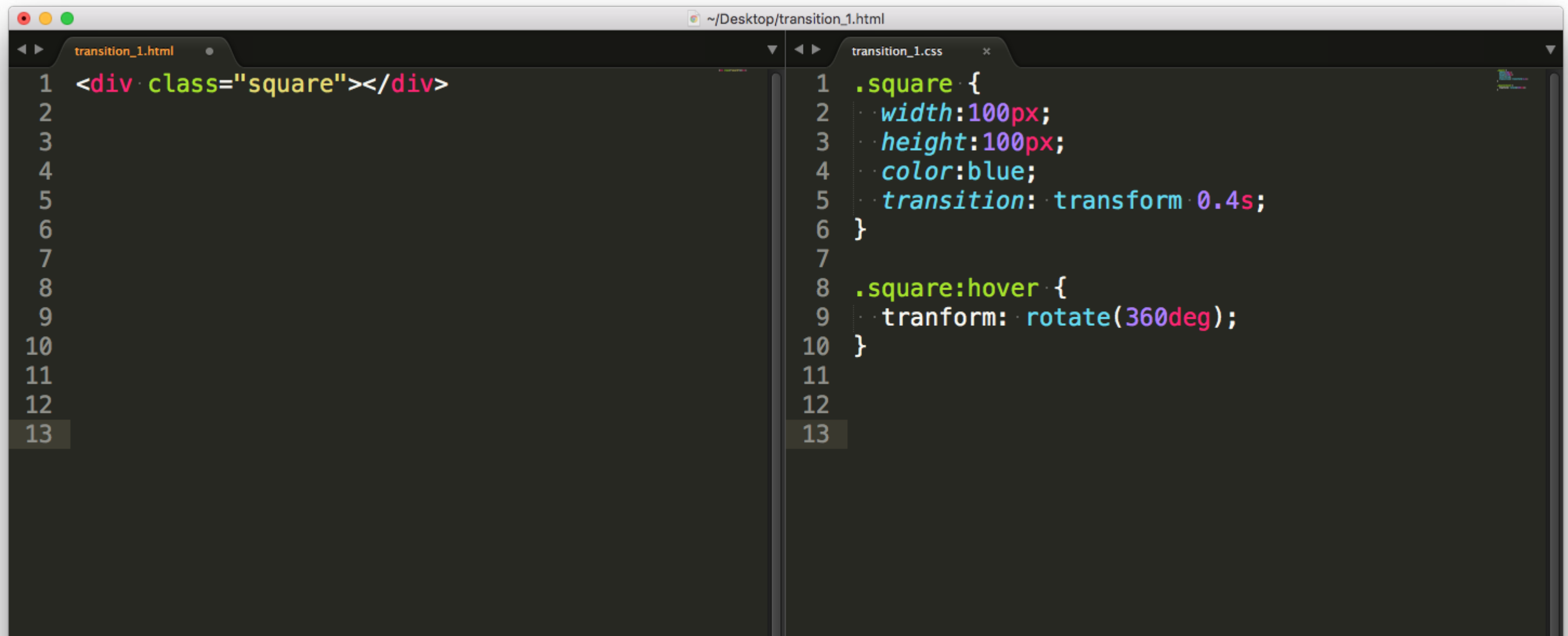
```
transform: rotate(360deg);
```

or

```
transform: rotate(1turn);
```

# Rotate + Transition

- Rotate still needs to happen over a duration! So you need to combine it with a transition to SEE it happen.



The image shows a code editor with two files open: `transition_1.html` and `transition_1.css`. The `transition_1.html` file contains a single line of HTML code: `<div class="square"></div>`. The `transition_1.css` file contains two CSS rules. The first rule, `.square {`, sets `width: 100px;`, `height: 100px;`, `color: blue;`, and `transition: transform 0.4s;`. The second rule, `.square:hover {`, sets `transform: rotate(360deg);`. The code is displayed in a dark-themed editor with syntax highlighting.

```
1 <div class="square"></div>
2
3
4
5
6
7
8
9
10
11
12
13

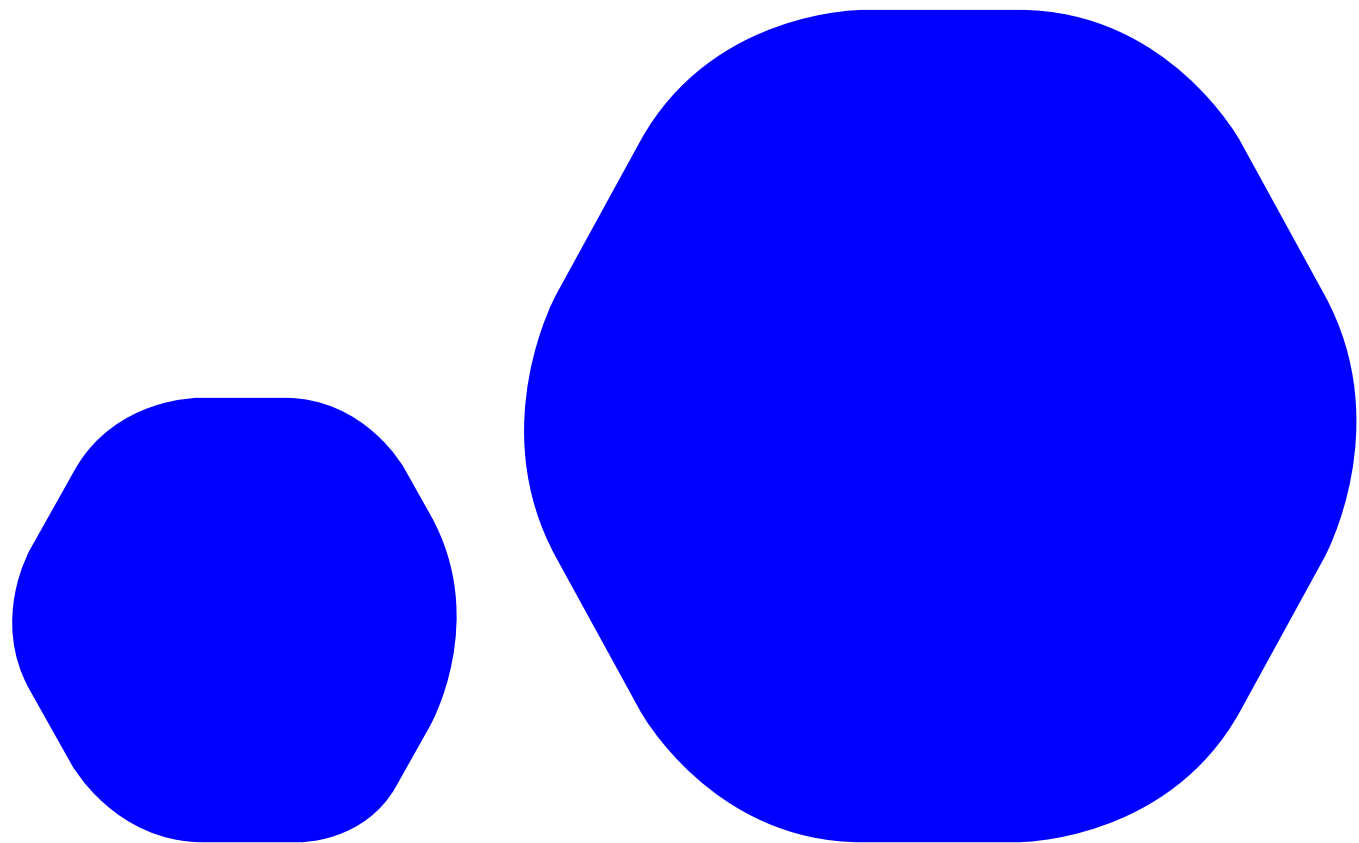
1 .square {
2   width: 100px;
3   height: 100px;
4   color: blue;
5   transition: transform 0.4s;
6 }
7
8 .square:hover {
9   transform: rotate(360deg);
10 }
11
12
13
```

# Transforming Scale

- Scale is used to stretch an element based on a value multiplier.

If only 1 value is provided, it will scale the element in both directions by that value:

```
transform: scale(2);
```



# Transforming Scale

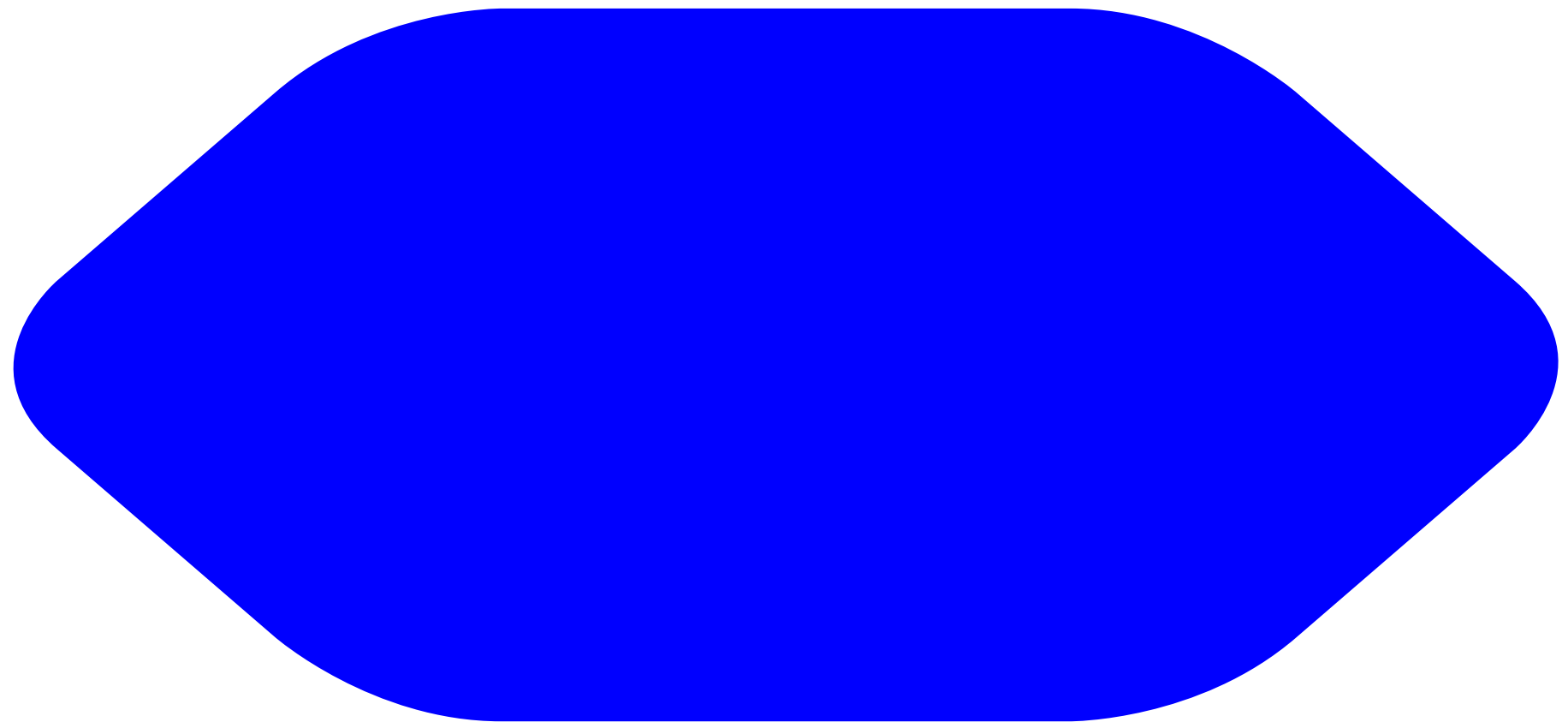
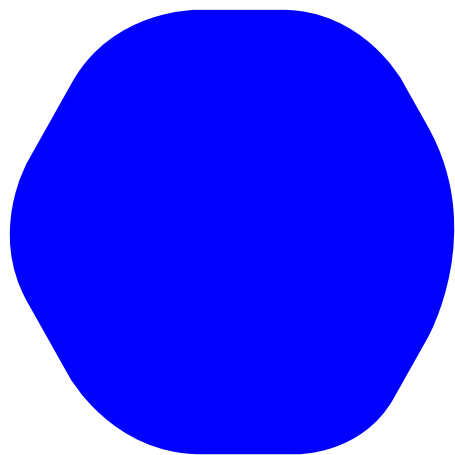
If only 2 value are provided, it will scale the element in the horizontal direction first, then the vertical direction.

```
transform: scale(4,2);
```

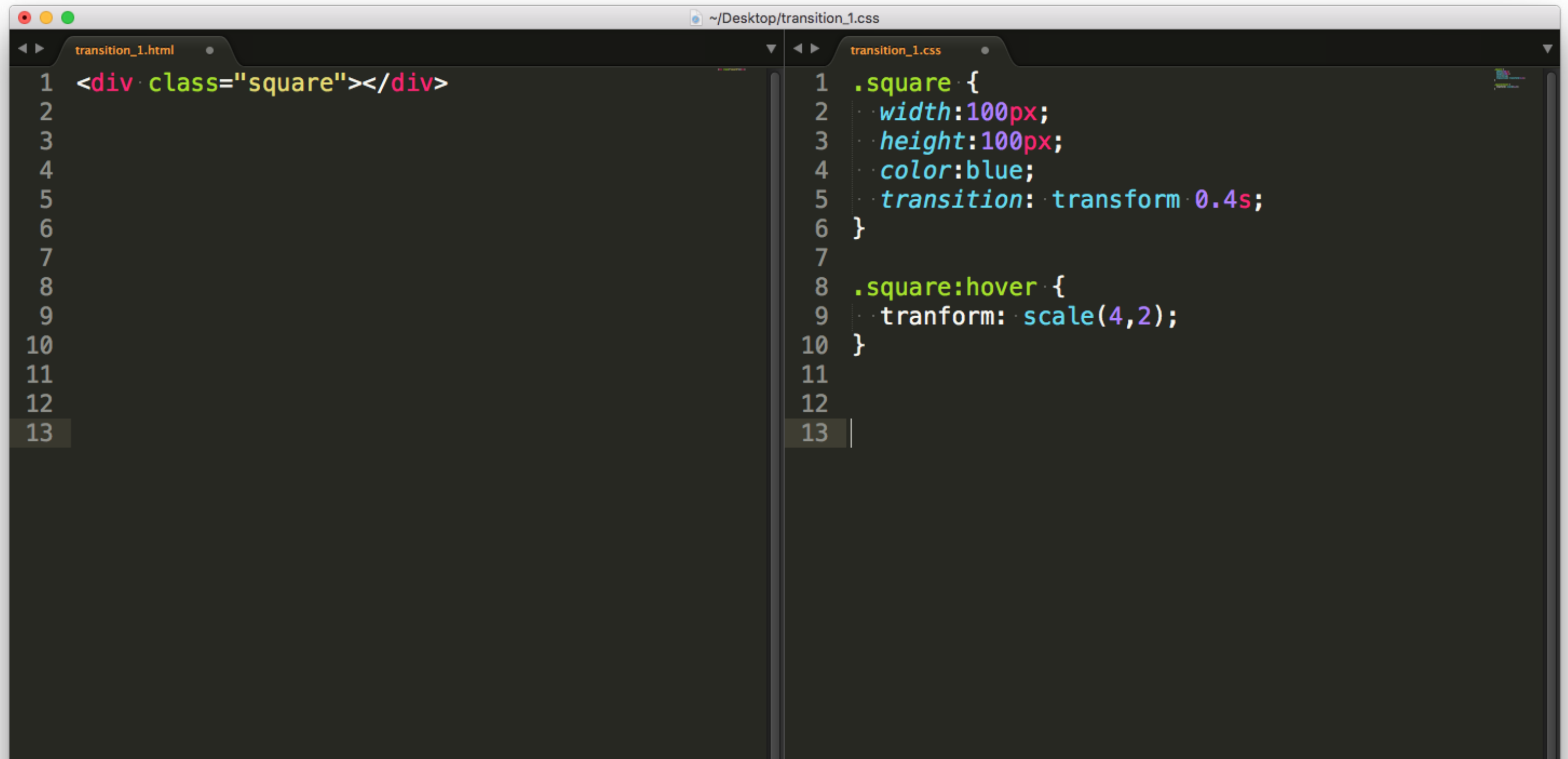
or

```
transform: scaleX(4);
```

```
transform: scaleY(2);
```



# Transforming Scale



The image shows a code editor window with two tabs: 'transition\_1.html' and 'transition\_1.css'. The 'transition\_1.html' tab is active, showing a single line of HTML code: `<div class="square"></div>`. The 'transition\_1.css' tab is also visible, showing CSS code for a square element. The CSS code includes a base style for the square and a hover state that scales the element.

```
1 <div class="square"></div>
2
3
4
5
6
7
8
9
10
11
12
13

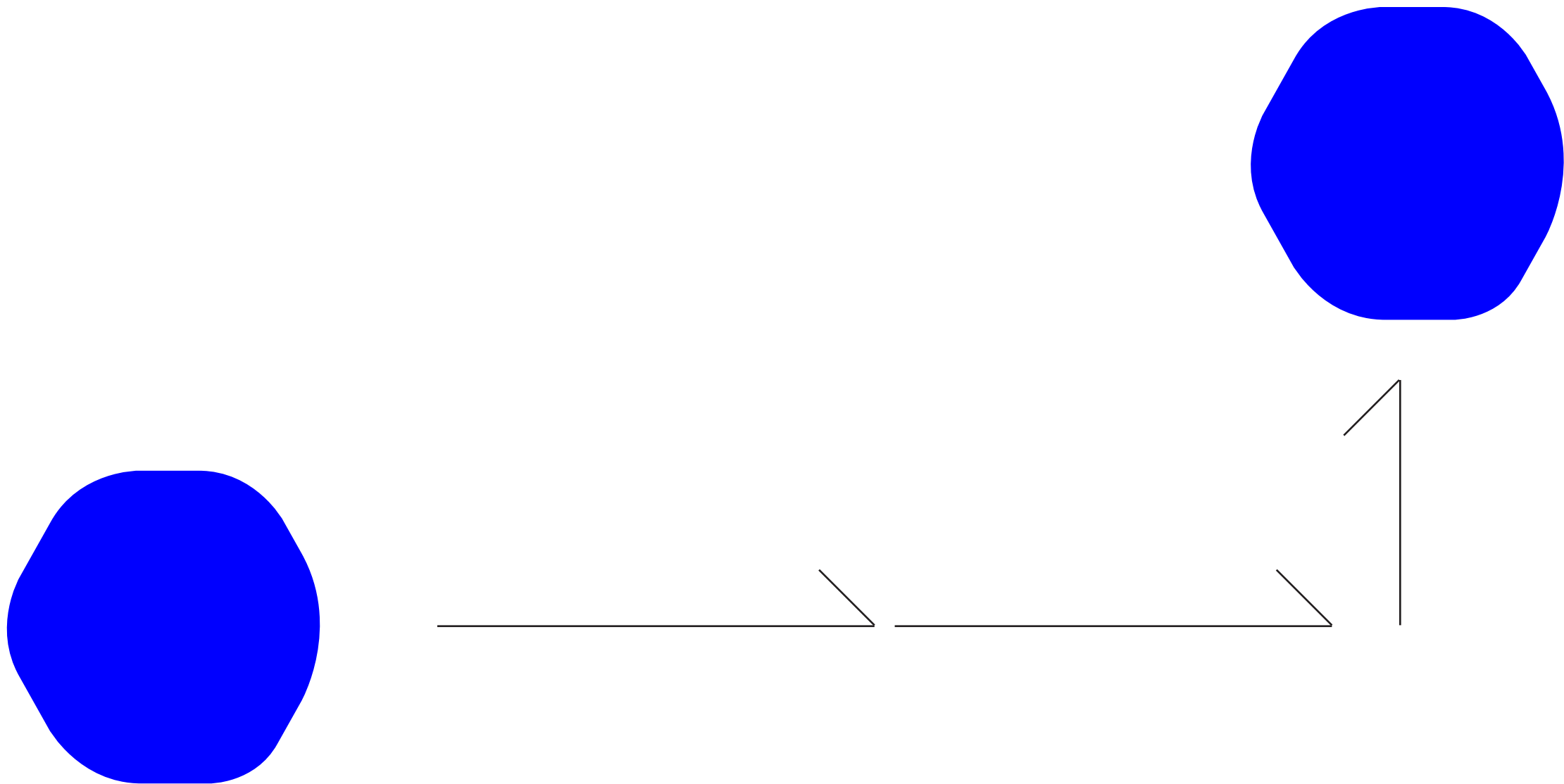
1 .square {
2   width: 100px;
3   height: 100px;
4   color: blue;
5   transition: transform 0.4s;
6 }
7
8 .square:hover {
9   transform: scale(4, 2);
10 }
11
12
13
```

# Transforming Translate

Translate simply means to move something.

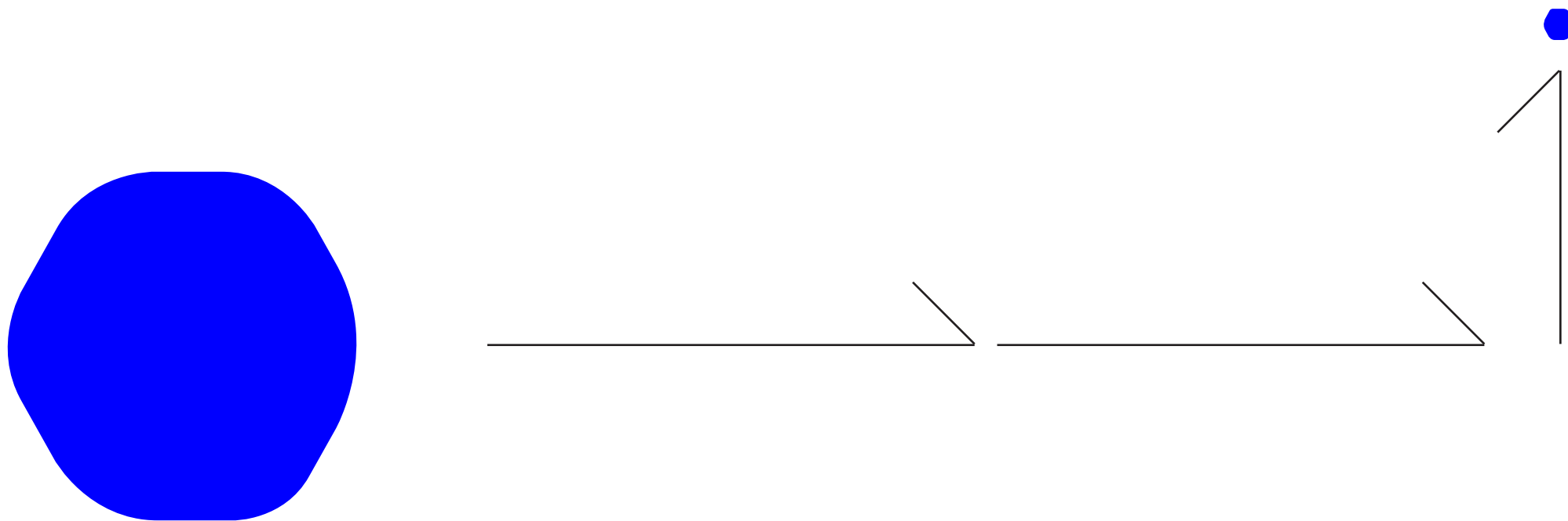
```
transform: translateX(500px);
```

```
transform: translateY(30px);
```



# Chaining

You can of course do many of these changes at once. For instance, you could scale down and translate to a new position.





# Next time

Keyframe animations

SVG animations