Cascading Style Sheet (CSS)

What is CSS?

- CSS stands for **C**ascading **S**tyle **S**heets
  - Styles define how to display HTML elements
  - Styles are normally stored in Style Sheets
  - External style sheets can save a lot of work
  - External style sheets are stored in CSS files
  - Multiple style definitions will cascade into one

  Example: http://www.csszengarden.com/

How to Insert a Style Sheet?

- External style sheet

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css" />
</head>
```

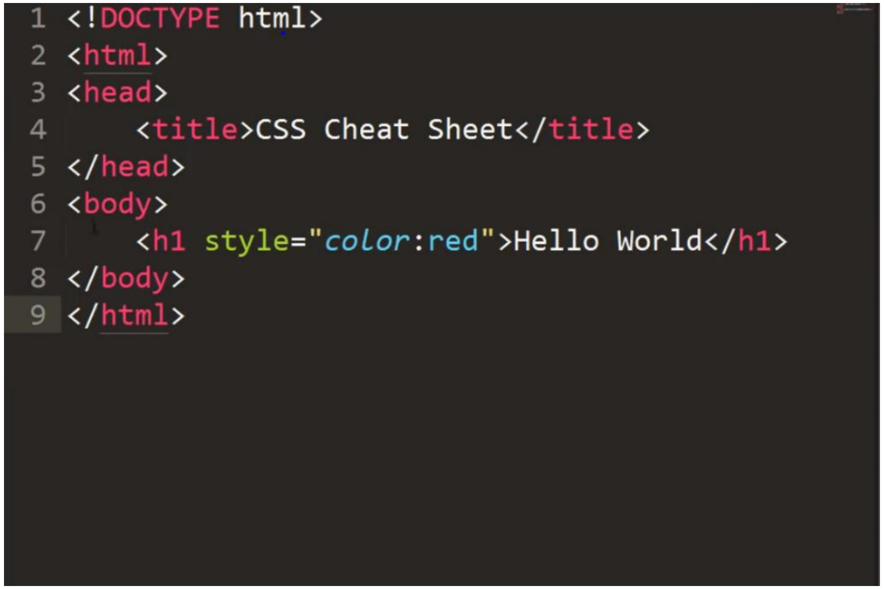
- Internal style sheet

```
<head>
<style type="text/css">
hr {color: sienna}
p {margin-left: 20px}
</style>
</head>
```

- Inline style

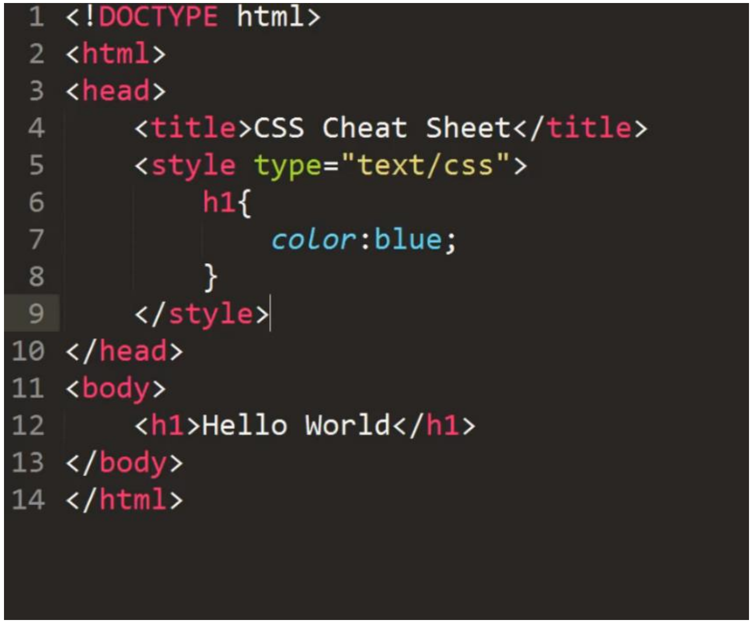
```
<p style="color: sienna; margin-left: 20px"> This is a paragraph </p>
```

Inline Styles



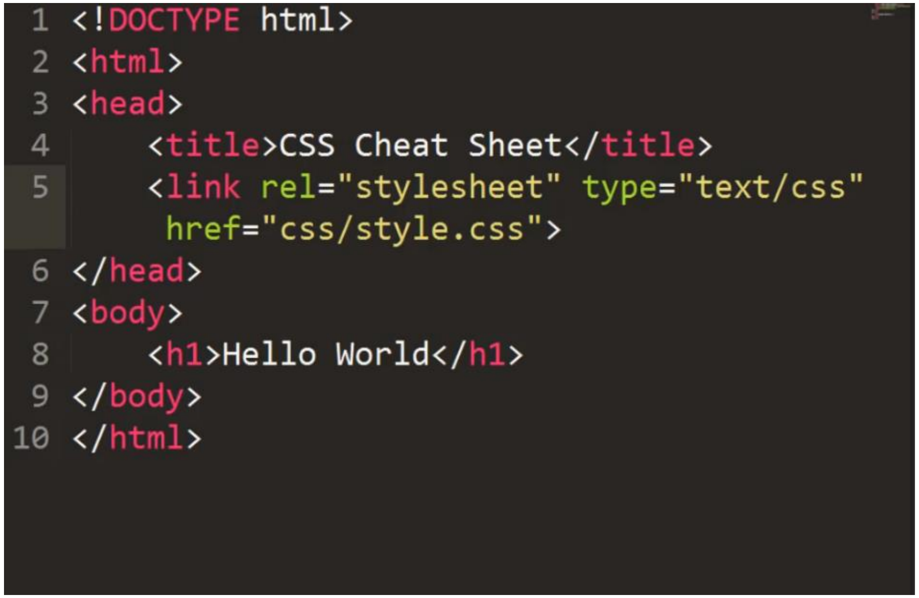
```html
<!DOCTYPE html>
<html>
<head>
    <title>CSS Cheat Sheet</title>
</head>
<body>
    <h1 style="color:red">Hello World</h1>
</body>
</html>
```

Internal Style Sheet



```
<!DOCTYPE html>
<html>
<head>
    <title>CSS Cheat Sheet</title>
    <style type="text/css">
        h1{
            color:blue;
        }
    </style>
</head>
<body>
    <h1>Hello World</h1>
</body>
</html>
```

External Style Sheet (.css files)

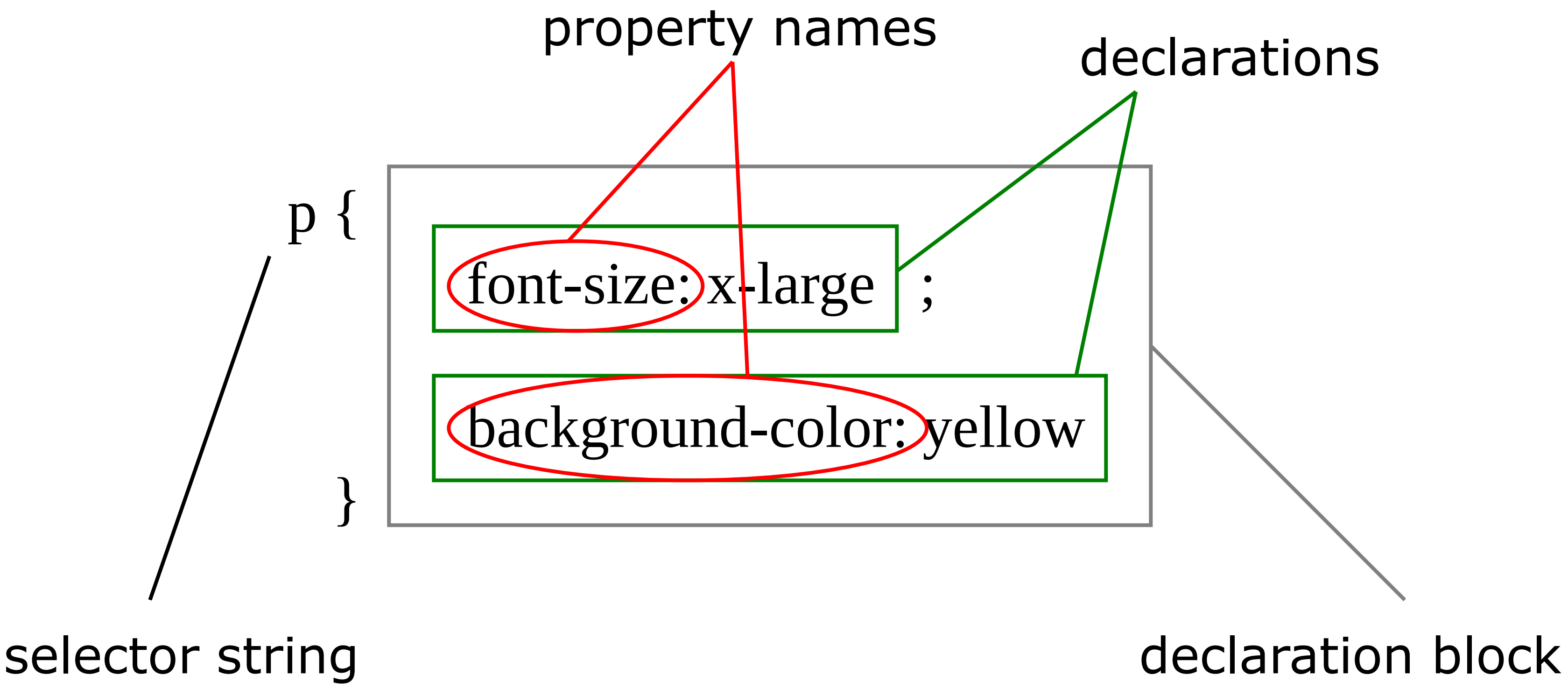


```
<!DOCTYPE html>
<html>
<head>
    <title>CSS Cheat Sheet</title>
    <link rel="stylesheet" type="text/css"
     href="css/style.css">
</head>
<body>
    <h1>Hello World</h1>
</body>
</html>
```

Multiple Styles Cascade Into One

- What style will be used when there is more than one style?
  - Browser default
  - External style sheets are included
  - Embedded styles (inside the <head> tag) override external styles
  - Inline styles (inside an HTML element) override both embedded and external styles
  - Styles modified with JavaScript override all other styles

CSS Style Rule



property names

declarations

```
p {
    font-size: x-large ;
    background-color: yellow
}
```

selector string

declaration block

Selector Strings

- Type selector:
  - Element type, such as body, p, hr, etc.
    - See previous example
  - Multiple element types using the same style are separated by comma
    - h1, h2, h3, h4, h5, h6 {background-color:purple}

- ID selector:
  - #p1, #s1 {background-color: blue}
  - <p id="p1"> … </p>
  - <span id="s1">…</span>
  - id values are case-sensitive

CSS Syntax

-
  ```
  p {
    color: red;
    text-align: center;
  }
  ```

-
  ```
  h1 {
    color: green;
    text-align: center;
  }
  ```

CSS Selectors

Selector	Example	Example description
#id	#firstname	Selects the element with id="firstname"
.class	.intro	Selects all elements with class="intro"
element.class	p.intro	Selects only <p> elements with class="intro"
*	*	Selects all elements
element	p	Selects all <p> elements
element,element,..	div, p	Selects all <div> elements and all <p> elements

- CSS Border Color
  - <h1 style="border:2px solid Violet;">Hello World</h1>

- CSS Text Color

  <h1 style="color:Tomato;">Hello World</h1>
  <p style="color:DodgerBlue;">Lorem ipsum...</p>
  <p style="color:MediumSeaGreen;">Ut wisi enim...</p>

CSS Background

- background-color
- background-image
- background-repeat
- background-attachment
- background-position
- background

Examples

-
  ```
  body {
    background-color: lightblue;
  }
  ```

-
  ```
  div {
    background-color: green;
    opacity: 0.3;
  }
  ```

-
  ```
  body {
    background-image: url("paper.gif");
  }
  ```

-
  ```
  p {
    background-image: url("paper.gif");
    background-repeat: repeat-x;
  ```

```
}
```

-
  ```
  body {
    background-image: url("img_tree.png");
    background-repeat: no-repeat;
  }
  ```

-
  ```
  body {
    background-image: url("img_tree.png");
    background-repeat: no-repeat;
    background-position: right top;
  }
  ```

-
  ```
  body {
    background-image: url("img_tree.png");
    background-repeat: no-repeat;
    background-position: right top;
    background-attachment: scroll;
  }
  ```

CSS Border Style

The following values are allowed:

- dotted - Defines a dotted border
- dashed - Defines a dashed border
- solid - Defines a solid border
- double - Defines a double border
- groove - Defines a 3D grooved border. The effect depends on the border-color value
- ridge - Defines a 3D ridged border. The effect depends on the border-color value
- inset - Defines a 3D inset border. The effect depends on the border-color value
- outset - Defines a 3D outset border. The effect depends on the border-color value
- none - Defines no border
- hidden - Defines a hidden border

-
  ```
  p {
    border-style: dotted solid;
  }
  ```

-
  ```
  p {
    border: 5px solid red;
  }
  ```

-
  ```
  p {
    border-left: 6px solid red;
    background-color: lightgrey;
  }
  ```

-
  ```
  p {
    border-bottom: 6px solid red;
    background-color: lightgrey;
  }
  ```

-
  ```
  p {
    border: 2px solid red;   %rounded border
  ```

CSS Margin

-
  ```
  p {
    margin-top: 100px;
    margin-bottom: 100px;
    margin-right: 150px;
    margin-left: 80px;
  }
  ```

-
  ```
  p {
    margin: 25px 50px 75px 100px;
  }
  ```

[top margin is 25px, right margin is 50px, bottom margin is 75px, left margin is 100px]

Property	Description
margin	A shorthand property for setting the margin properties in one declaration
margin-bottom	Sets the bottom margin of an element
margin-left	Sets the left margin of an element
margin-right	Sets the right margin of an element
margin-top	Sets the top margin of an element

CSS Padding

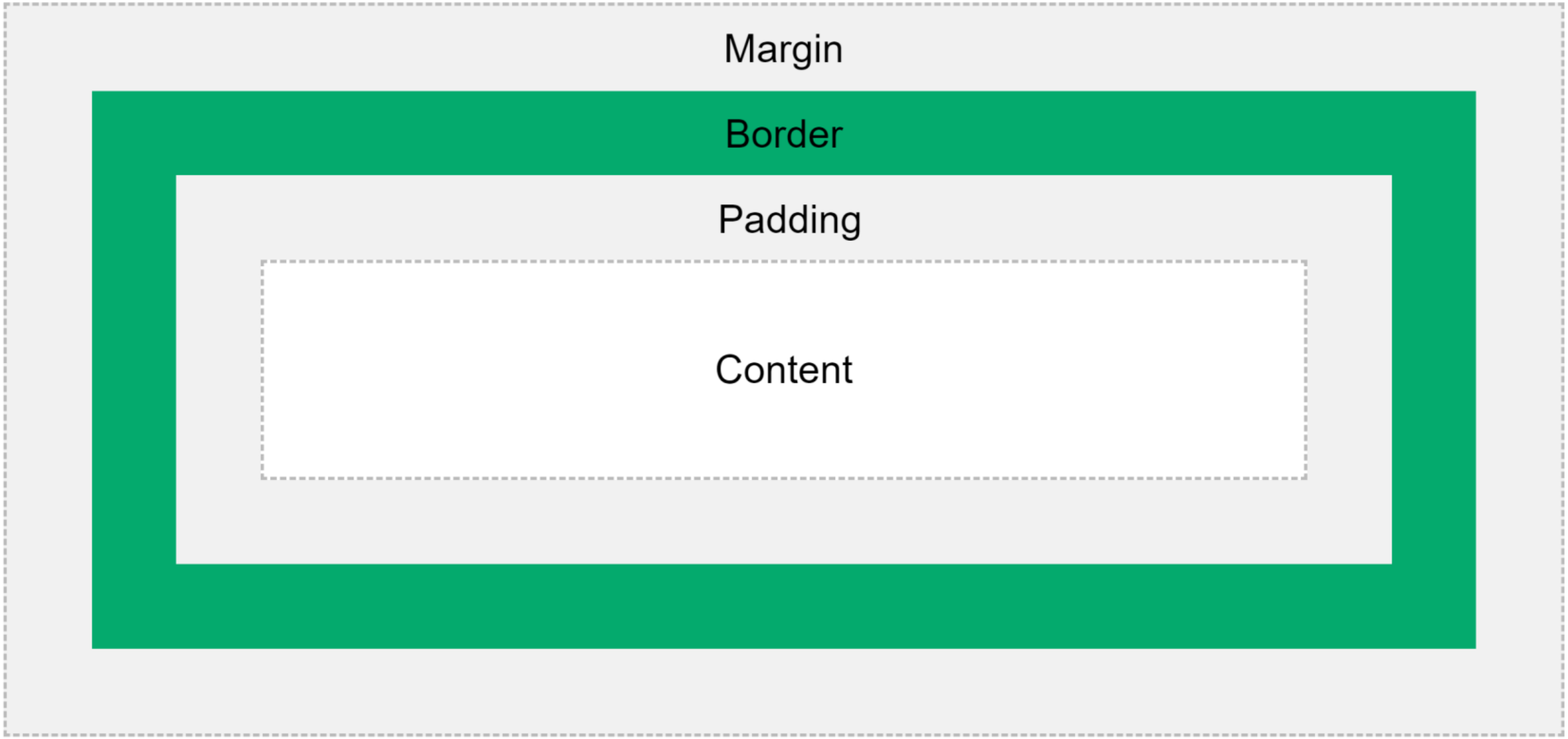
.The CSS `padding` properties are used to generate space around an element's content, inside of any defined borders.

.With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

-
  ```
  div {
    padding-top: 50px;
    padding-right: 30px;
    padding-bottom: 50px;
    padding-left: 80px;
  }
  ```

-
  ```
  div {
    padding: 25px 50px 75px 100px;
  }
  ```

[top padding is 25px, right padding is 50px, bottom padding is 75px, left padding is 100px]


Margin
Border
Padding
Content

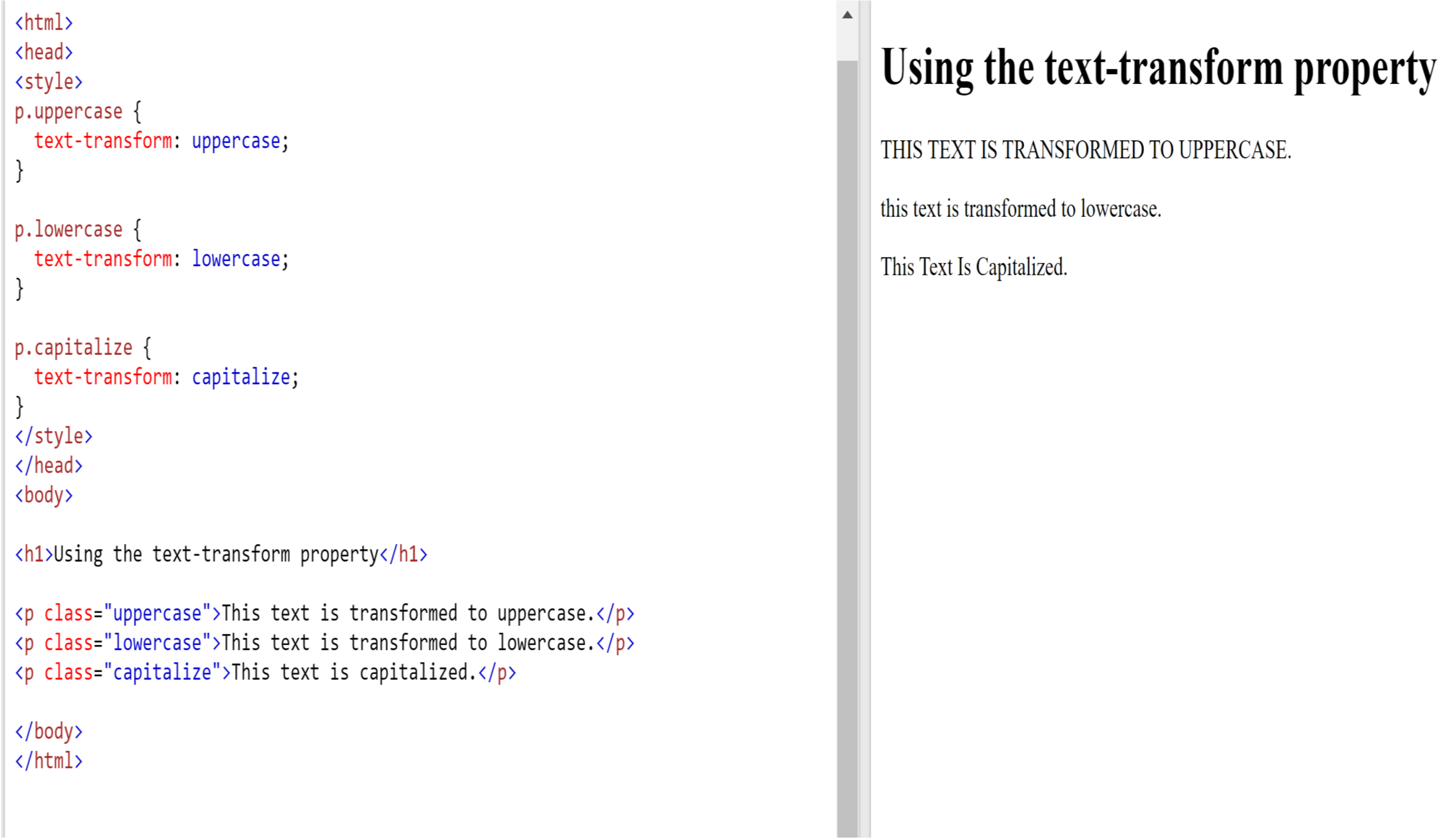
Text

-
  ```
  div {
    text-align: justify;
  }
  ```

-
  ```
  h1 {
    letter-spacing: 5px;
  }
  ```

-
  ```
  h1 {
    text-shadow: 2px 2px red;
  }
  ```

-
  ```
  p {
  font-family: Tahoma, Verdana, sans-serif;
  }
  ```



```
<html>
<head>
<style>
p.uppercase {
  text-transform: uppercase;
}

p.lowercase {
  text-transform: lowercase;
}

p.capitalize {
  text-transform: capitalize;
}
</style>
</head>
<body>

<h1>Using the text-transform property</h1>

<p class="uppercase">This text is transformed to uppercase.</p>
<p class="lowercase">This text is transformed to lowercase.</p>
<p class="capitalize">This text is capitalized.</p>

</body>
</html>
```

Using the text-transform property

THIS TEXT IS TRANSFORMED TO UPPERCASE.

this text is transformed to lowercase.

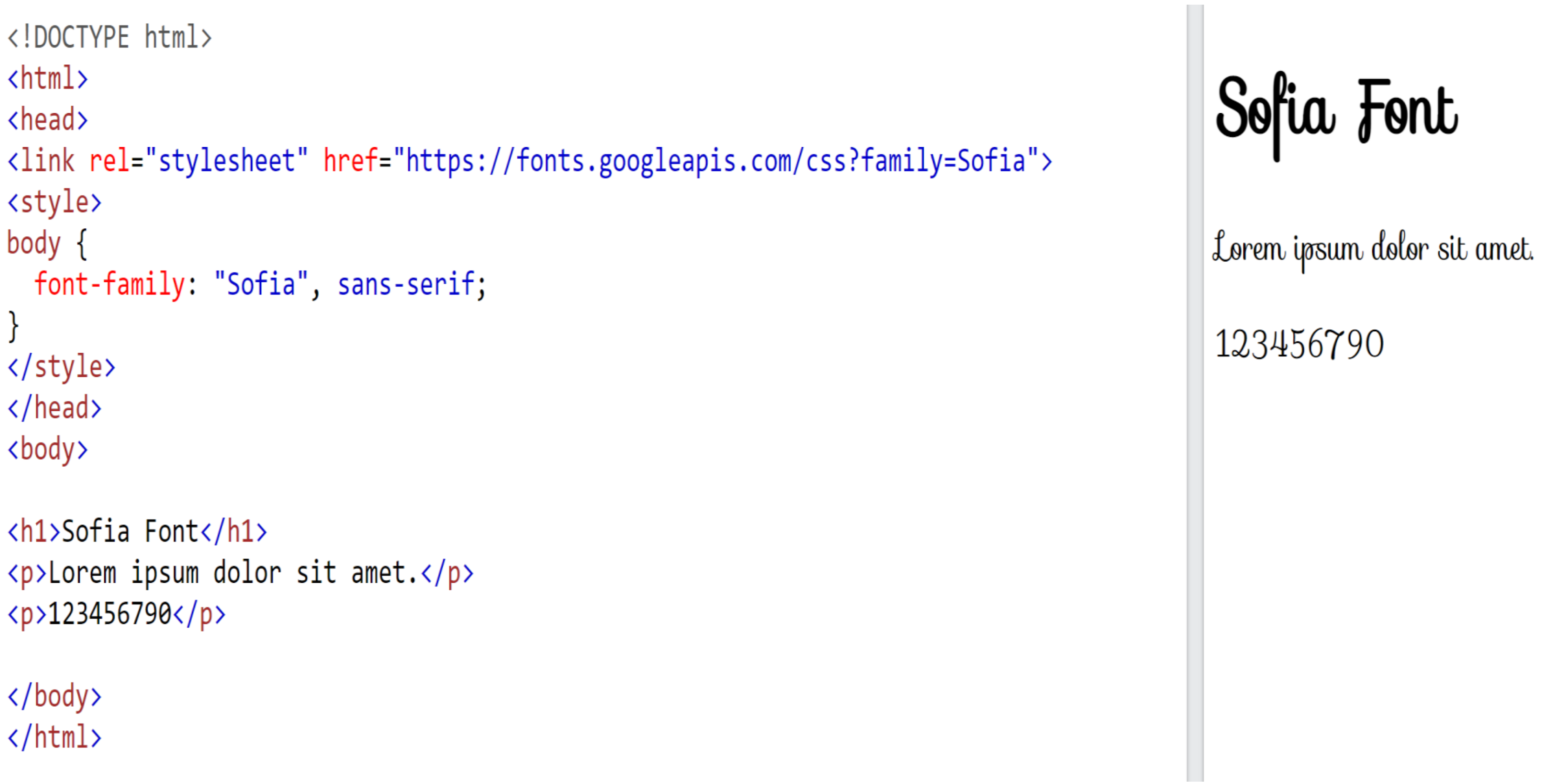
This Text Is Capitalized.

-
  ```
  p.normal {
        font-style: normal;
  }
  ```

-
  ```
  p.normal {
    font-weight: normal;
  }
  ```

-
  ```
  h1 {
    font-size: 40px;
  }
  ```

-
  ```
  <p class="normal">This is a paragraph in
  normal style.</p>
  ```

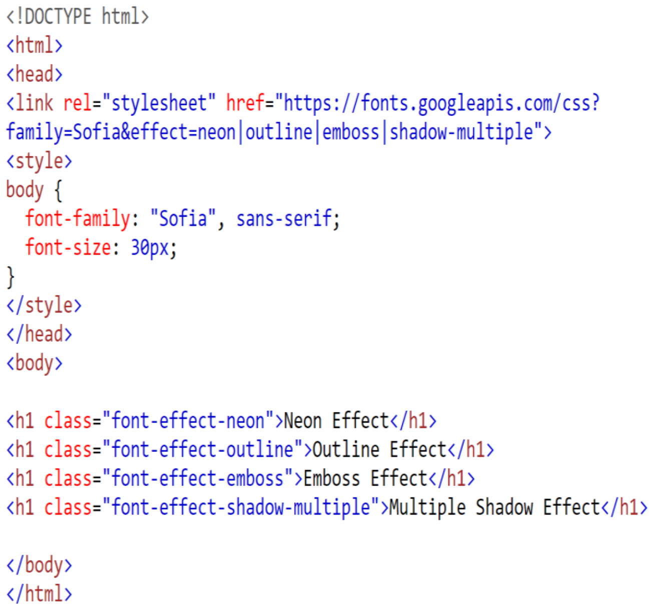


```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia">
<style>
body {
  font-family: "Sofia", sans-serif;
}
</style>
</head>
<body>

<h1>Sofia Font</h1>
<p>Lorem ipsum dolor sit amet.</p>
<p>123456790</p>

</body>
</html>
```

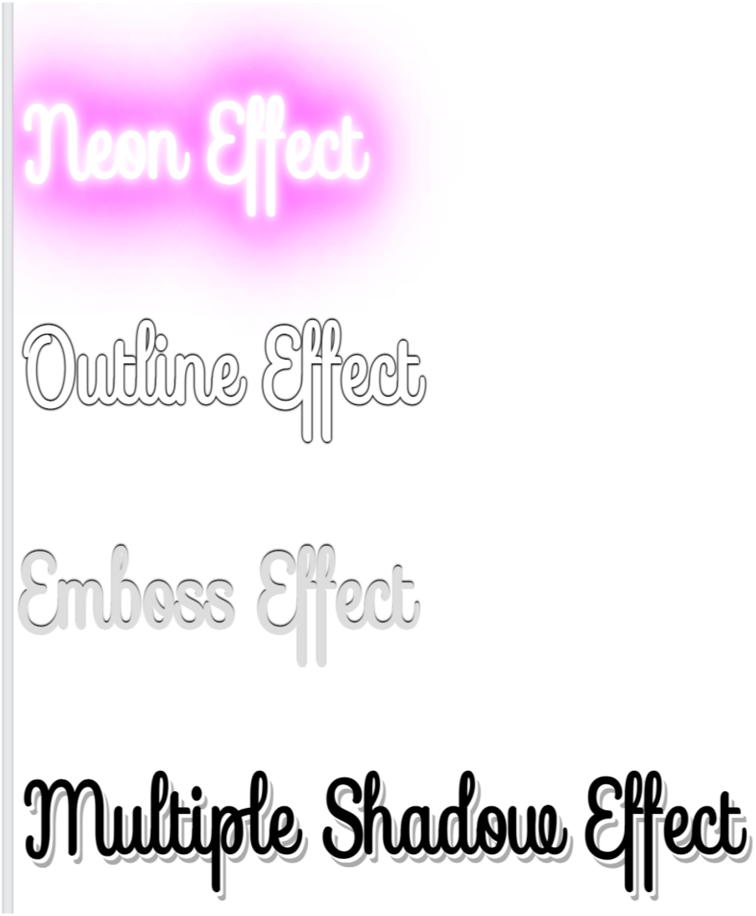
Sofia Font

Lorem ipsum dolor sit amet.

123456790



```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?
family=Sofia&effect=neon|outline|emboss|shadow-multiple">
<style>
body {
  font-family: "Sofia", sans-serif;
  font-size: 30px;
}
</style>
</head>
<body>

<h1 class="font-effect-neon">Neon Effect</h1>
<h1 class="font-effect-outline">Outline Effect</h1>
<h1 class="font-effect-emboss">Emboss Effect</h1>
<h1 class="font-effect-shadow-multiple">Multiple Shadow Effect</h1>

</body>
</html>
```



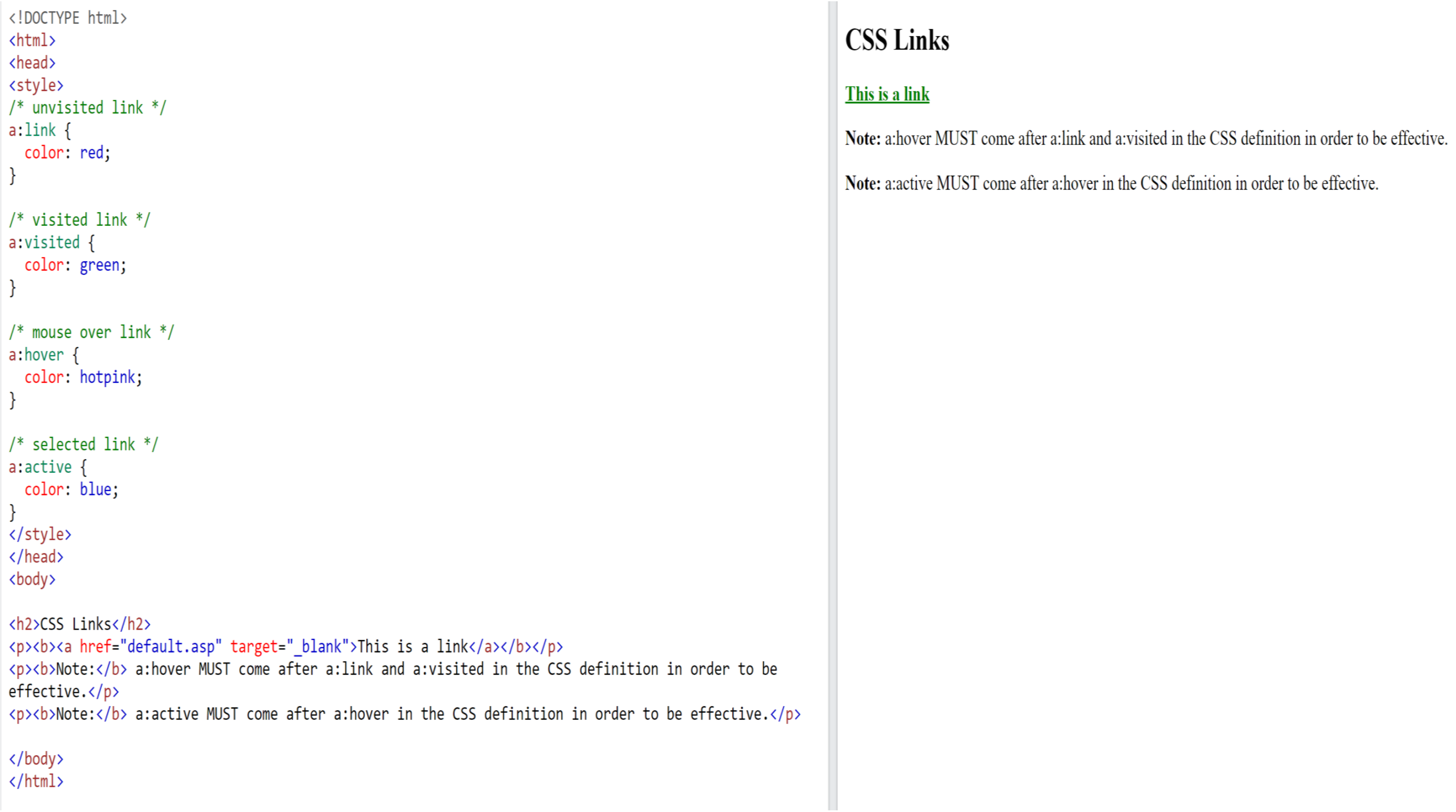
Neon Effect

Outline Effect

Emboss Effect

Multiple Shadow Effect

- Font pairing
- Font shorthand



```
<!DOCTYPE html>
<html>
<head>
<style>
/* unvisited link */
a:link {
  color: red;
}

/* visited link */
a:visited {
  color: green;
}

/* mouse over link */
a:hover {
  color: hotpink;
}

/* selected link */
a:active {
  color: blue;
}
</style>
</head>
<body>

<h2>CSS Links</h2>
<p><b><a href="default.asp" target="_blank">This is a link</a></b></p>
<p><b>Note:</b> a:hover MUST come after a:link and a:visited in the CSS definition in order to be
effective.</p>
<p><b>Note:</b> a:active MUST come after a:hover in the CSS definition in order to be effective.</p>

</body>
</html>
```

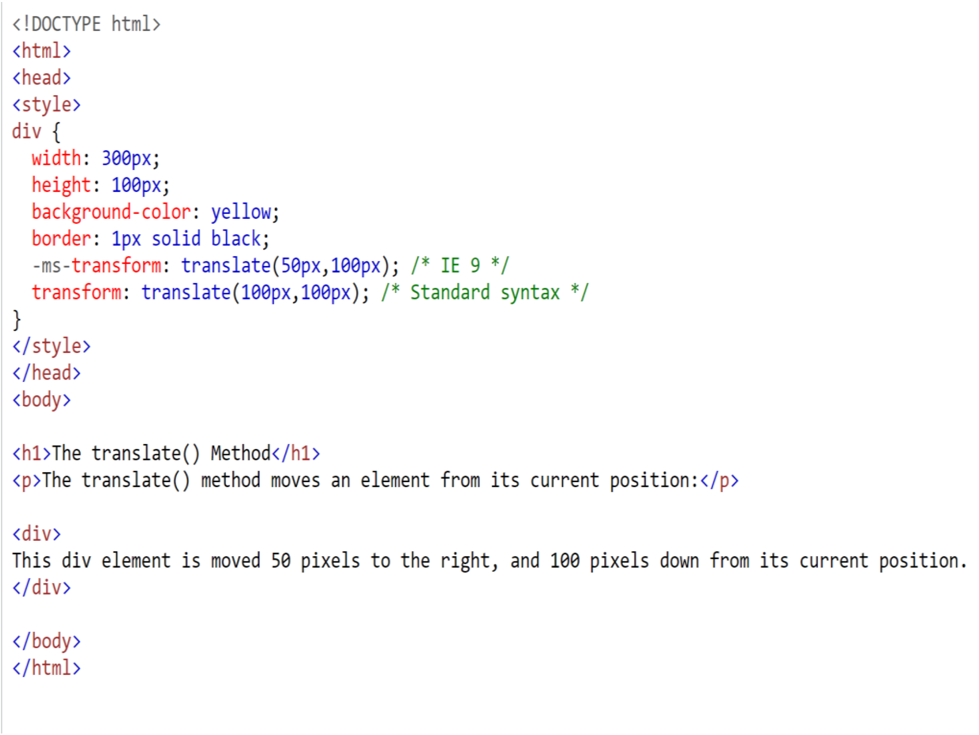
CSS Links

This is a link

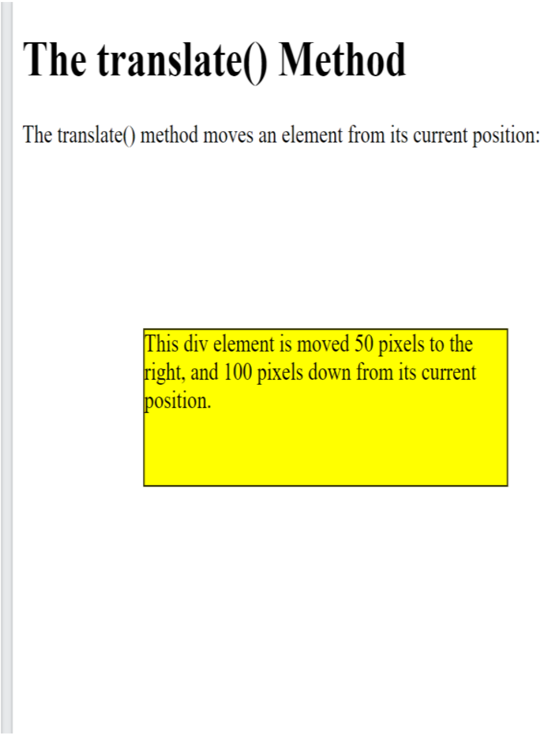
Note: a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.

Note: a:active MUST come after a:hover in the CSS definition in order to be effective.

Translate method



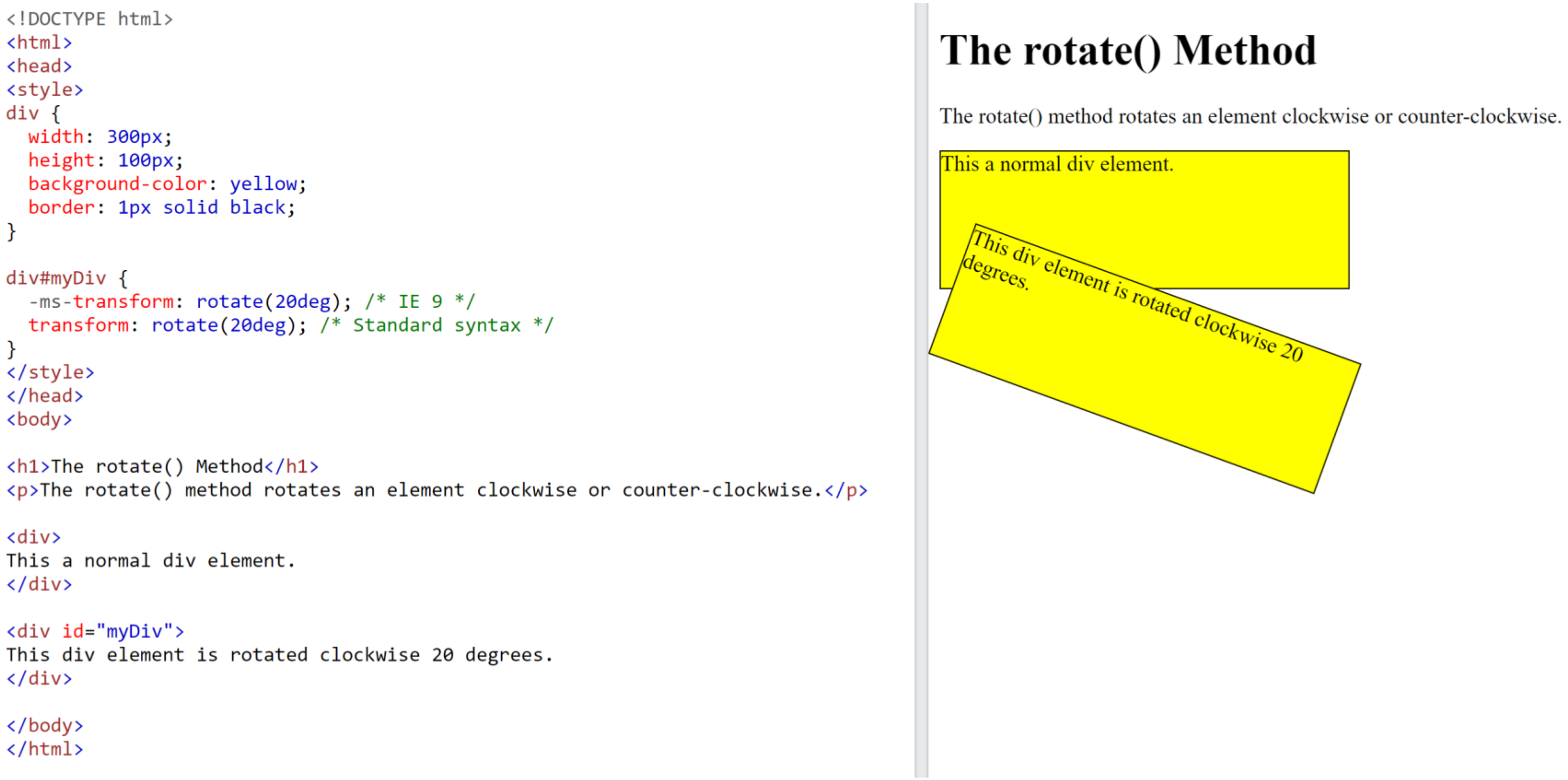
```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 300px;
  height: 100px;
  background-color: yellow;
  border: 1px solid black;
  -ms-transform: translate(50px,100px); /* IE 9 */
  transform: translate(100px,100px); /* Standard syntax */
}
</style>
</head>
<body>

<h1>The translate() Method</h1>
<p>The translate() method moves an element from its current position:</p>

<div>
This div element is moved 50 pixels to the right, and 100 pixels down from its current position.
</div>

</body>
</html>
```



The translate() Method

The translate() method moves an element from its current position:

This div element is moved 50 pixels to the right, and 100 pixels down from its current position.



```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 300px;
  height: 100px;
  background-color: yellow;
  border: 1px solid black;
}

div#myDiv {
  -ms-transform: rotate(20deg); /* IE 9 */
  transform: rotate(20deg); /* Standard syntax */
}
</style>
</head>
<body>

<h1>The rotate() Method</h1>
<p>The rotate() method rotates an element clockwise or counter-clockwise.</p>

<div>
This a normal div element.
</div>

<div id="myDiv">
This div element is rotated clockwise 20 degrees.
</div>

</body>
</html>
```

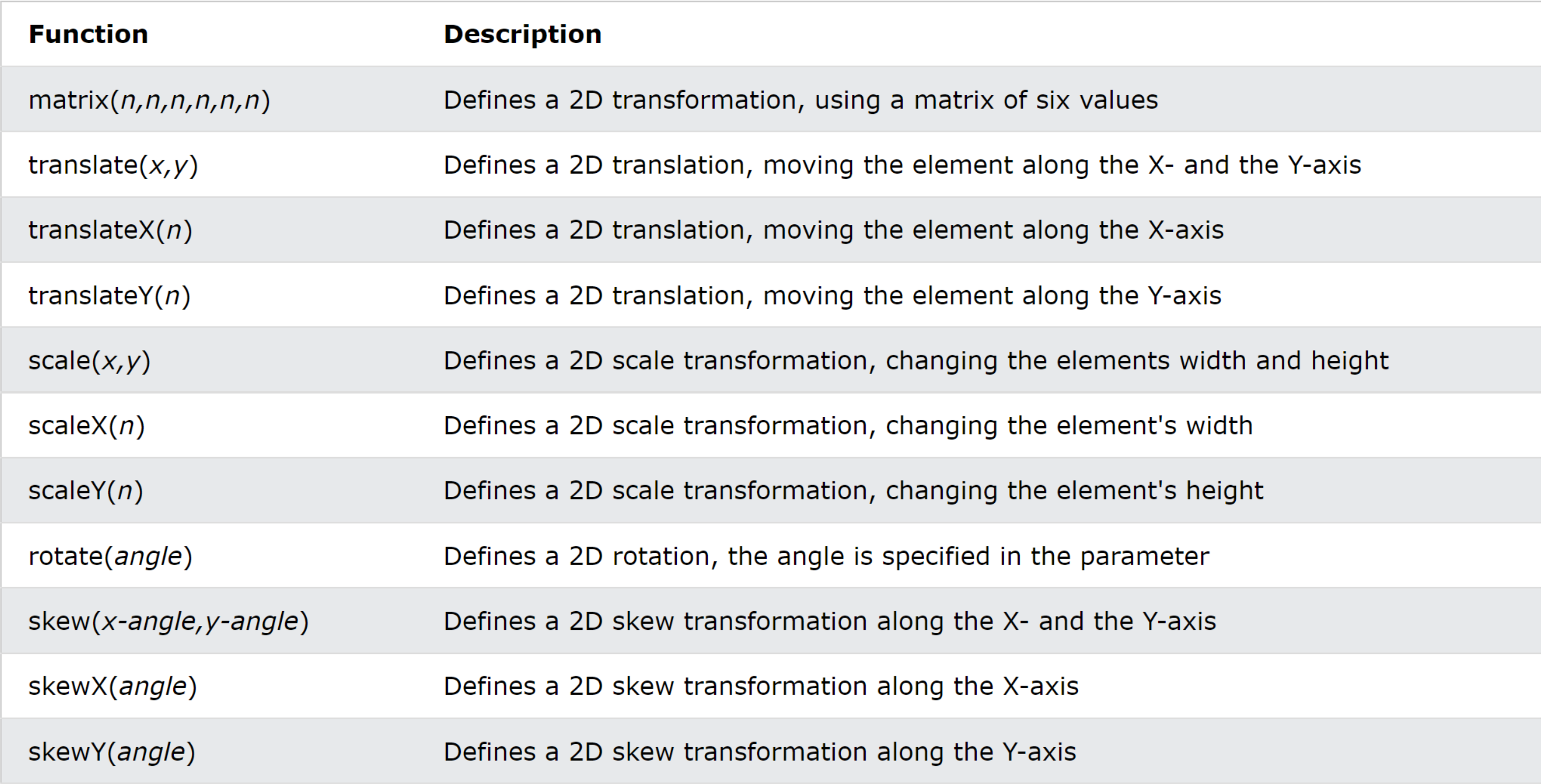
The rotate() Method

The rotate() method rotates an element clockwise or counter-clockwise.

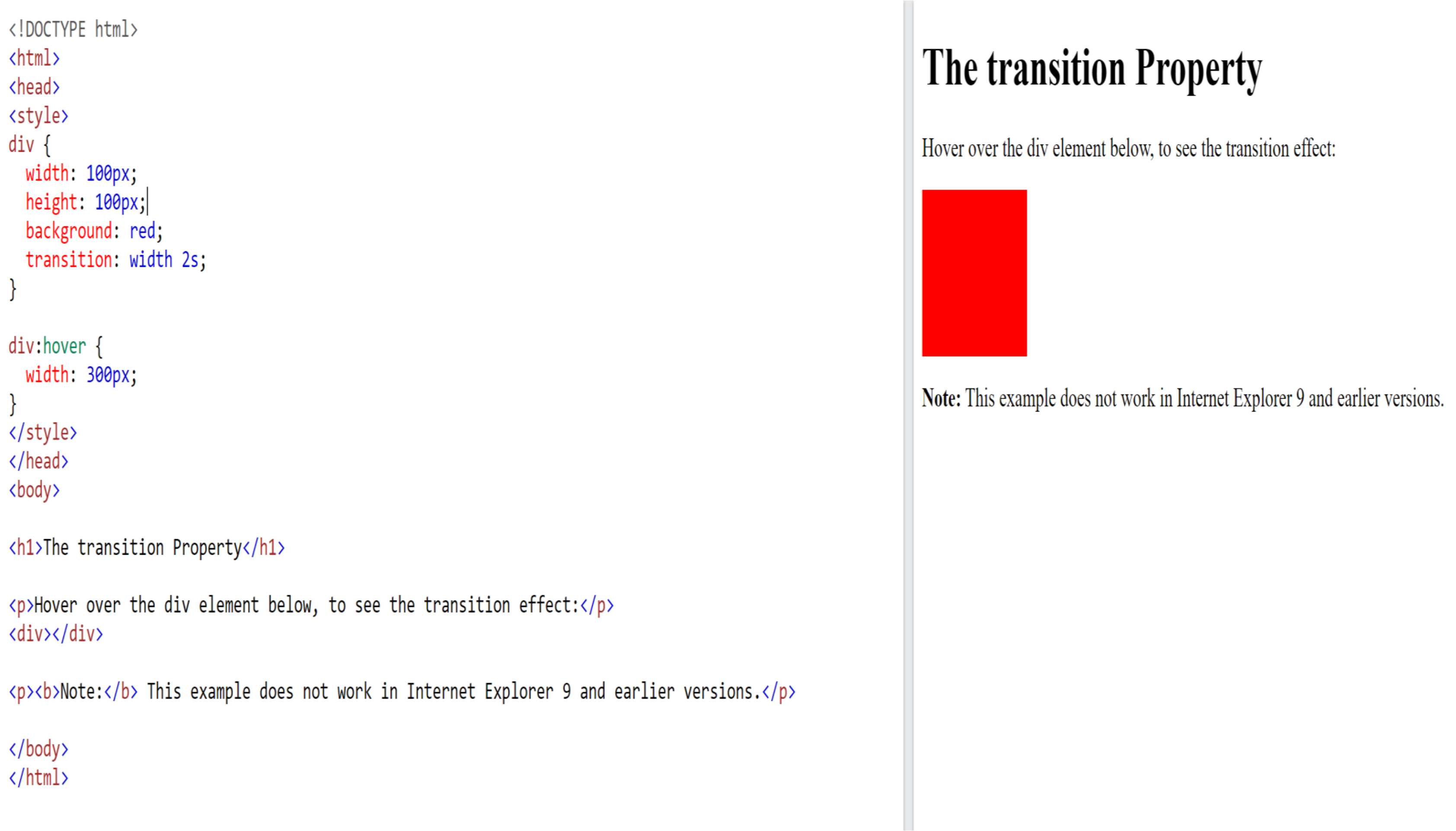
This a normal div element.

This div element is rotated clockwise 20 degrees.

CSS 2D Transform Methods



Function	Description
matrix(*n,n,n,n,n,n*)	Defines a 2D transformation, using a matrix of six values
translate(*x,y*)	Defines a 2D translation, moving the element along the X- and the Y-axis
translateX(*n*)	Defines a 2D translation, moving the element along the X-axis
translateY(*n*)	Defines a 2D translation, moving the element along the Y-axis
scale(*x,y*)	Defines a 2D scale transformation, changing the elements width and height
scaleX(*n*)	Defines a 2D scale transformation, changing the element's width
scaleY(*n*)	Defines a 2D scale transformation, changing the element's height
rotate(*angle*)	Defines a 2D rotation, the angle is specified in the parameter
skew(*x-angle,y-angle*)	Defines a 2D skew transformation along the X- and the Y-axis
skewX(*angle*)	Defines a 2D skew transformation along the X-axis
skewY(*angle*)	Defines a 2D skew transformation along the Y-axis

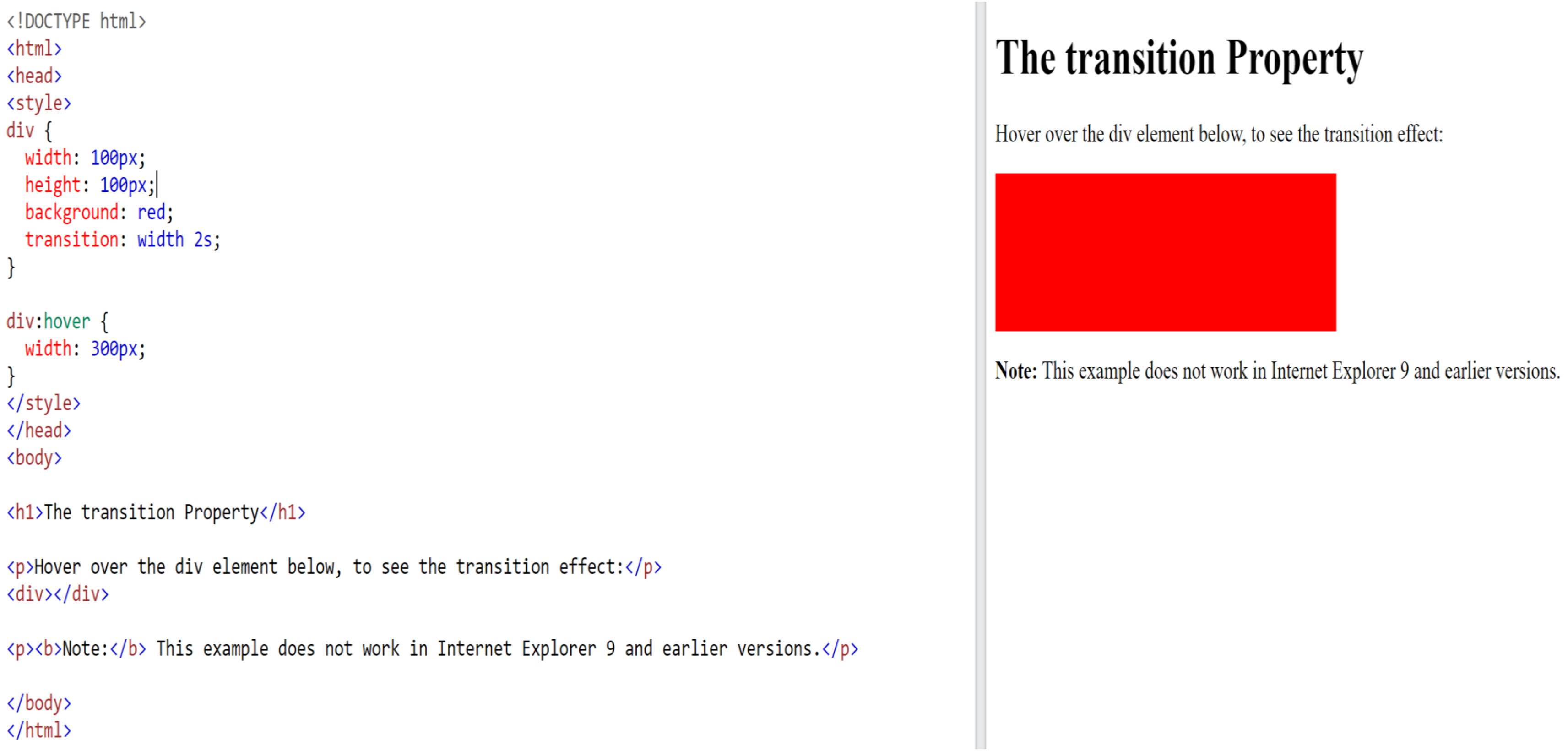


```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s;
}

div:hover {
  width: 300px;
}
</style>
</head>
<body>

<h1>The transition Property</h1>

<p>Hover over the div element below, to see the transition effect:</p>
<div></div>

<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>

</body>
</html>
```

The transition Property

Hover over the div element below, to see the transition effect:

Note: This example does not work in Internet Explorer 9 and earlier versions.



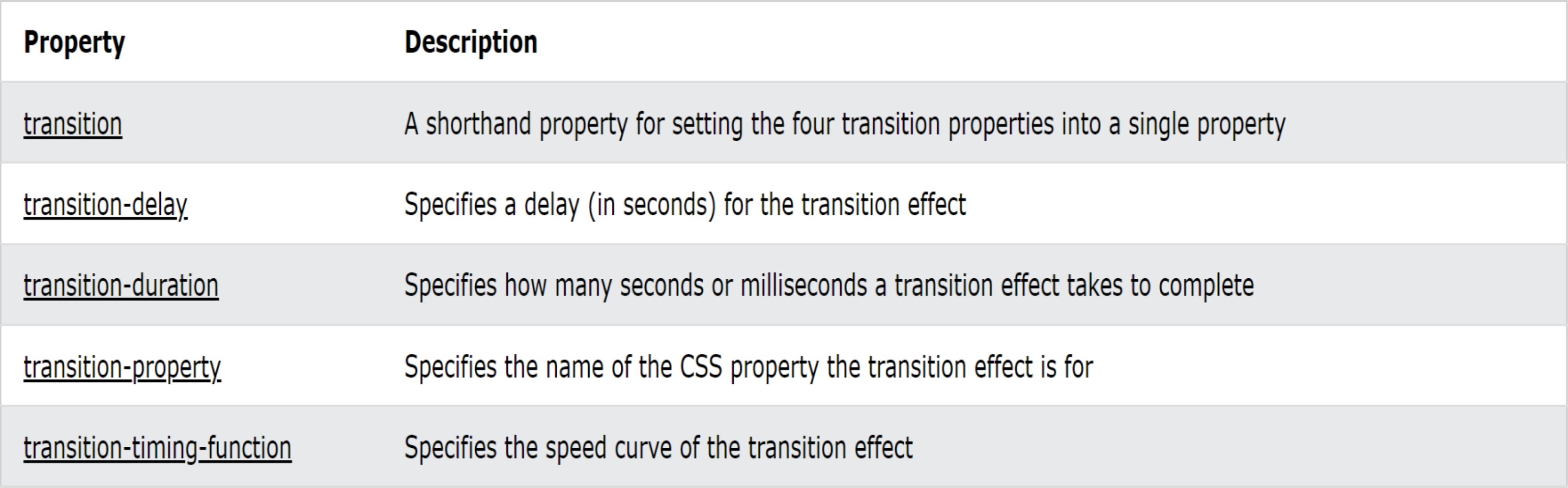
```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s;
}

div:hover {
  width: 300px;
}
</style>
</head>
<body>

<h1>The transition Property</h1>

<p>Hover over the div element below, to see the transition effect:</p>
<div></div>

<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>

</body>
</html>
```

The transition Property

Hover over the div element below, to see the transition effect:

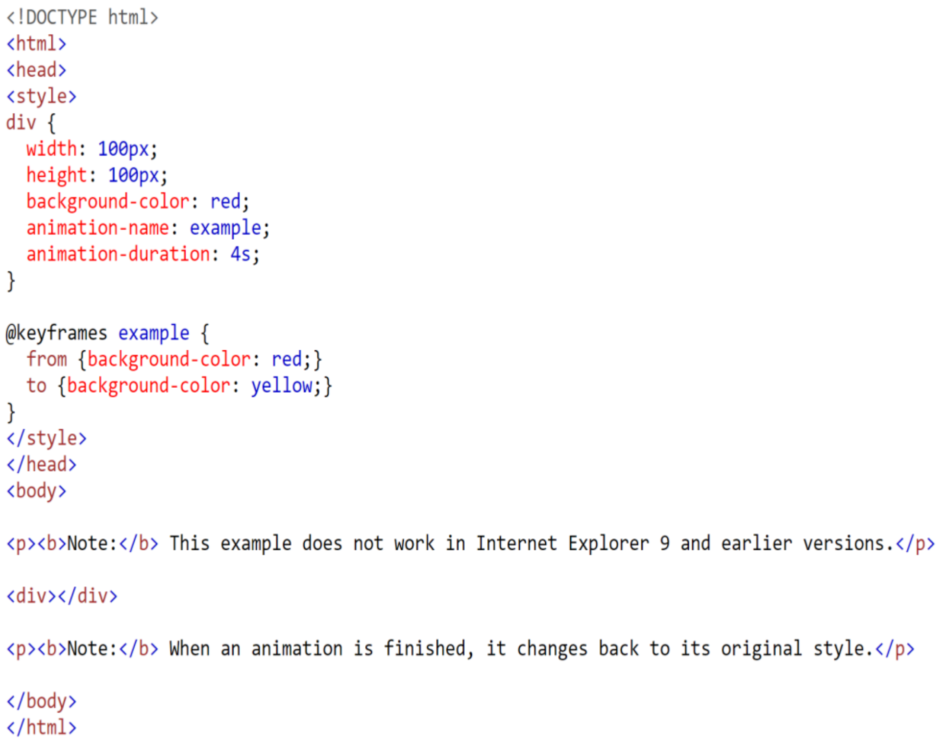
Note: This example does not work in Internet Explorer 9 and earlier versions.

Transition Property

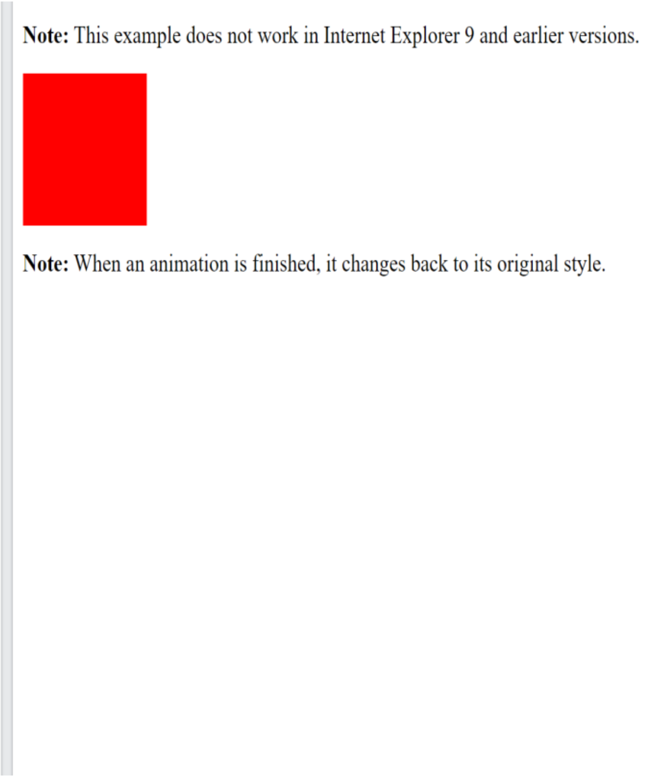


Property	Description
transition	A shorthand property for setting the four transition properties into a single property
transition-delay	Specifies a delay (in seconds) for the transition effect
transition-duration	Specifies how many seconds or milliseconds a transition effect takes to complete
transition-property	Specifies the name of the CSS property the transition effect is for
transition-timing-function	Specifies the speed curve of the transition effect

CSS Animation



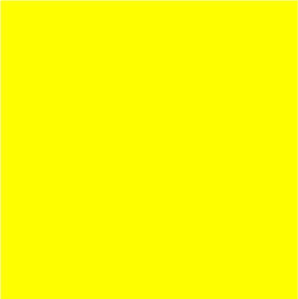
```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}

@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}
</style>
</head>
<body>

<p><b>Note:</b> This example does not work in Internet Explorer 9 and earlier versions.</p>

<div></div>

<p><b>Note:</b> When an animation is finished, it changes back to its original style.</p>

</body>
</html>
```



Note: This example does not work in Internet Explorer 9 and earlier versions.

Note: When an animation is finished, it changes back to its original style.

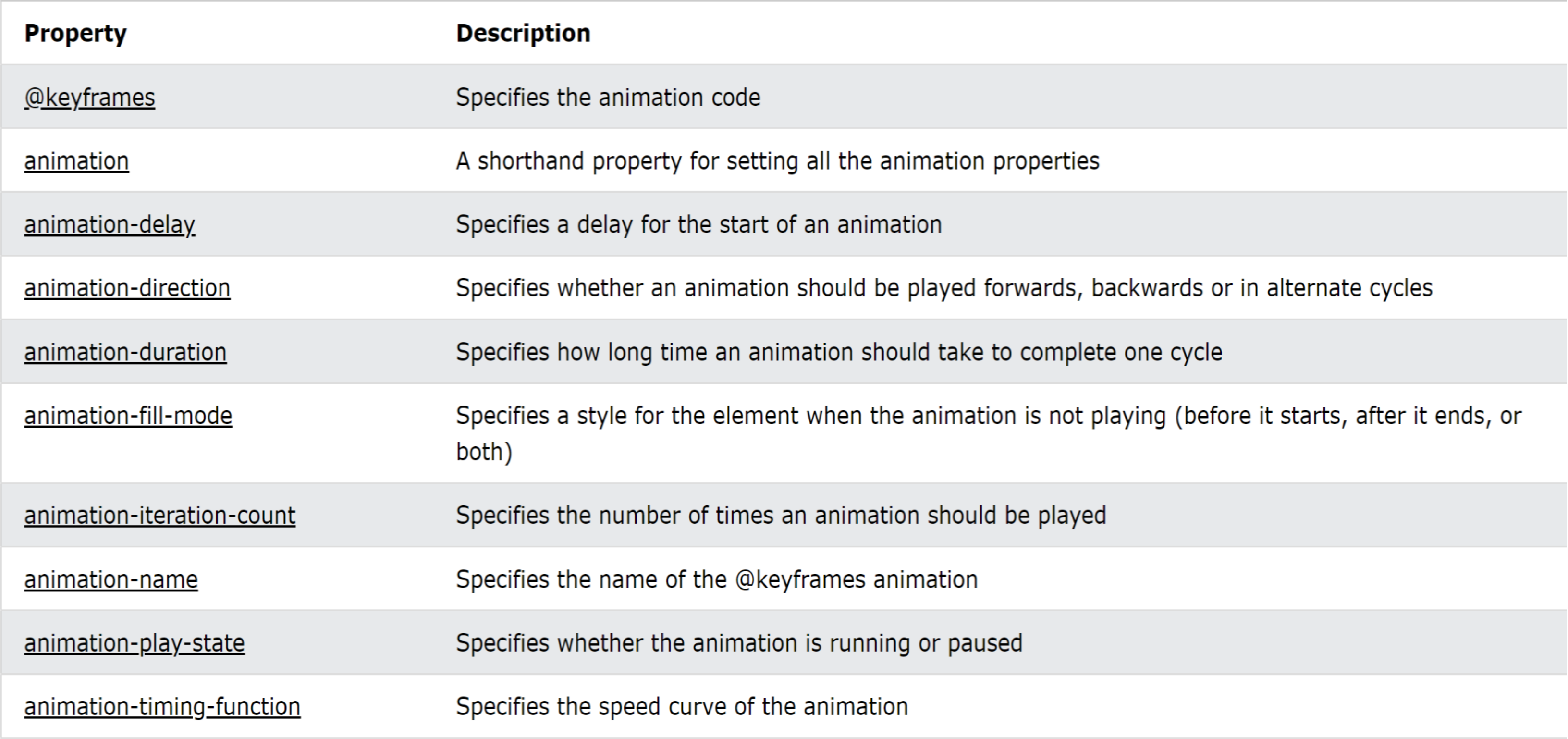
Note: This example does not work in Internet Explorer 9 and earlier versions.



Note: When an animation is finished, it changes back to its original style.

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

  /* The element to apply the animation to */
  div {
    width: 100px;
    height: 100px;
    background-color: red;
    animation-name: example;
    animation-duration: 4s;
  }
  ```

- Animation Delay
- Animation Count
- Animation Direction



Property	Description
@keyframes	Specifies the animation code
animation	A shorthand property for setting all the animation properties
animation-delay	Specifies a delay for the start of an animation
animation-direction	Specifies whether an animation should be played forwards, backwards or in alternate cycles
animation-duration	Specifies how long time an animation should take to complete one cycle
animation-fill-mode	Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both)
animation-iteration-count	Specifies the number of times an animation should be played
animation-name	Specifies the name of the @keyframes animation
animation-play-state	Specifies whether the animation is running or paused
animation-timing-function	Specifies the speed curve of the animation

ToolTip

- A tooltip is often used to specify extra information about something when the user moves the mouse pointer over an element:
  - Bottom
  - Right
  - Left

CSS Styling Images

- Rounded image

```
 img {
  border-radius: 8px;
}
```

- Circled image

```
 img {
  border-radius: 50%;
}
```

- Thumbnail image with link

```
 img {
  border: 1px solid #ddd;
  border-radius: 4px;
  padding: 5px;
  width: 150px;
}

img:hover {
  box-shadow: 0 0 2px 1px rgba(0, 140, 186, 0.5);
}

<a href="paris.jpg">
  <img src="paris.jpg" alt="Paris">
</a>
```

- Responsive image: If you want an image to scale down if it has to, but never scale up to be larger than its original size, add the following:

```
img {
  max-width: 100%;
  height: auto;
}
```

- Center an Image
- Image opacity
- Image text
- Image filter

- Image Reflection

```
img {
  -webkit-box-reflect: right;
}
```

- Image Reflection with Gradient
- Object fit

Object-fit

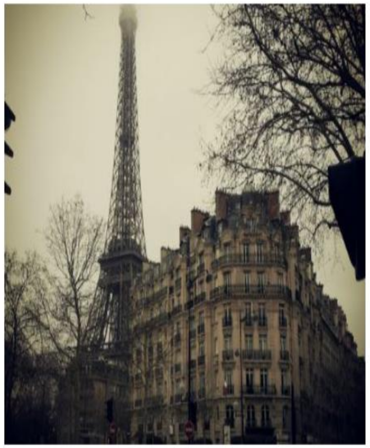
- The CSS object-fit property is used to specify how an <img> or <video> should be resized to fit its container.

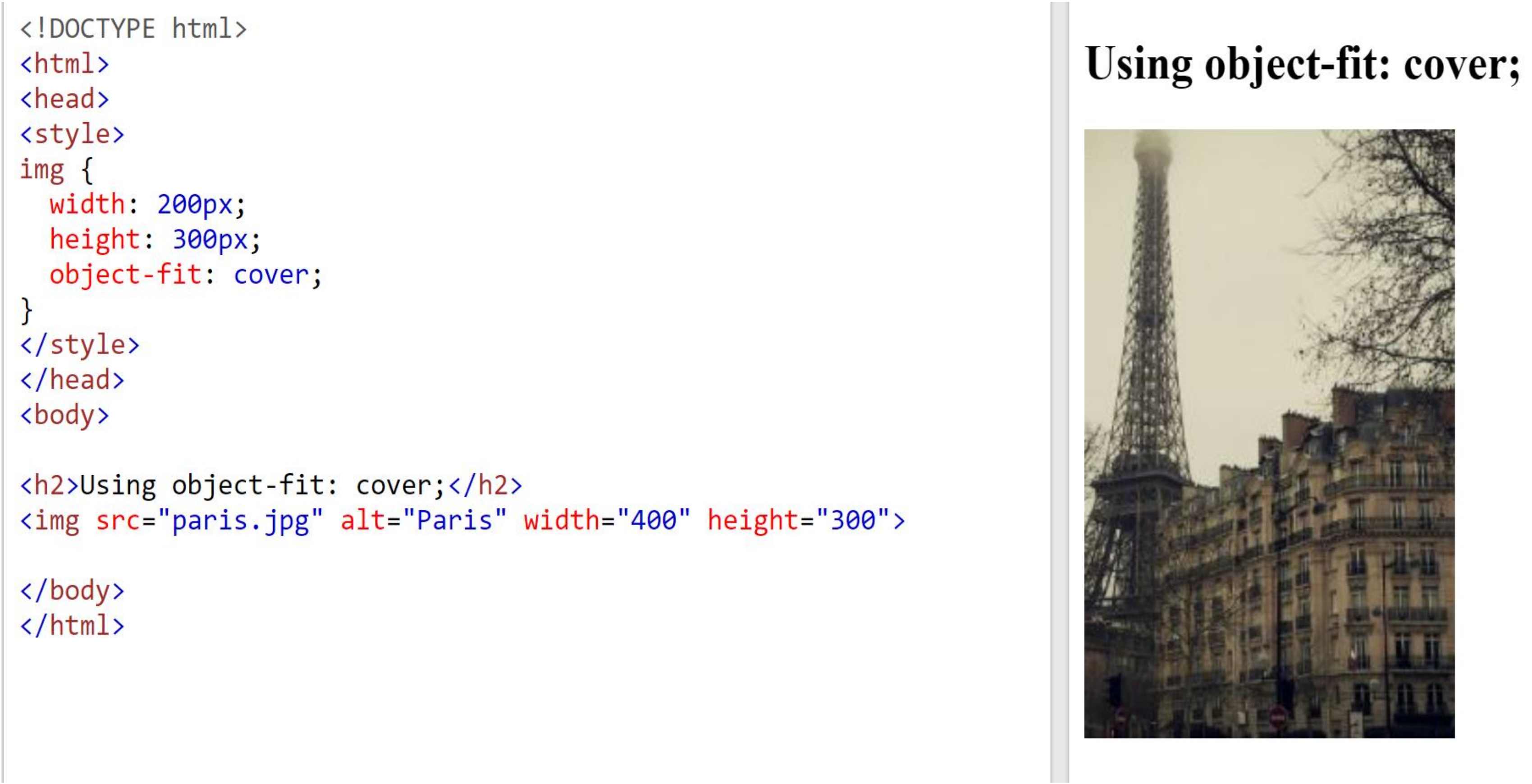
The object-fit property can take one of the following values:

  - **Fill**: This is default. The image is resized to fill given dimension. If necessary, the image will be stretched or squished to fit
  - **Contain**: The image keeps its aspect ratio, but is resized to fit within the given dimension
  - **Cover**: The image keeps its aspect ratio and fills the given dimension. The image will be clipped to fit
  - **None**: The image is not resized
  - **scale-down**: the image is scaled down to the smallest version of none or contain

Look at the following image from Paris. This image is 400 pixels wide and 300 pixels high:

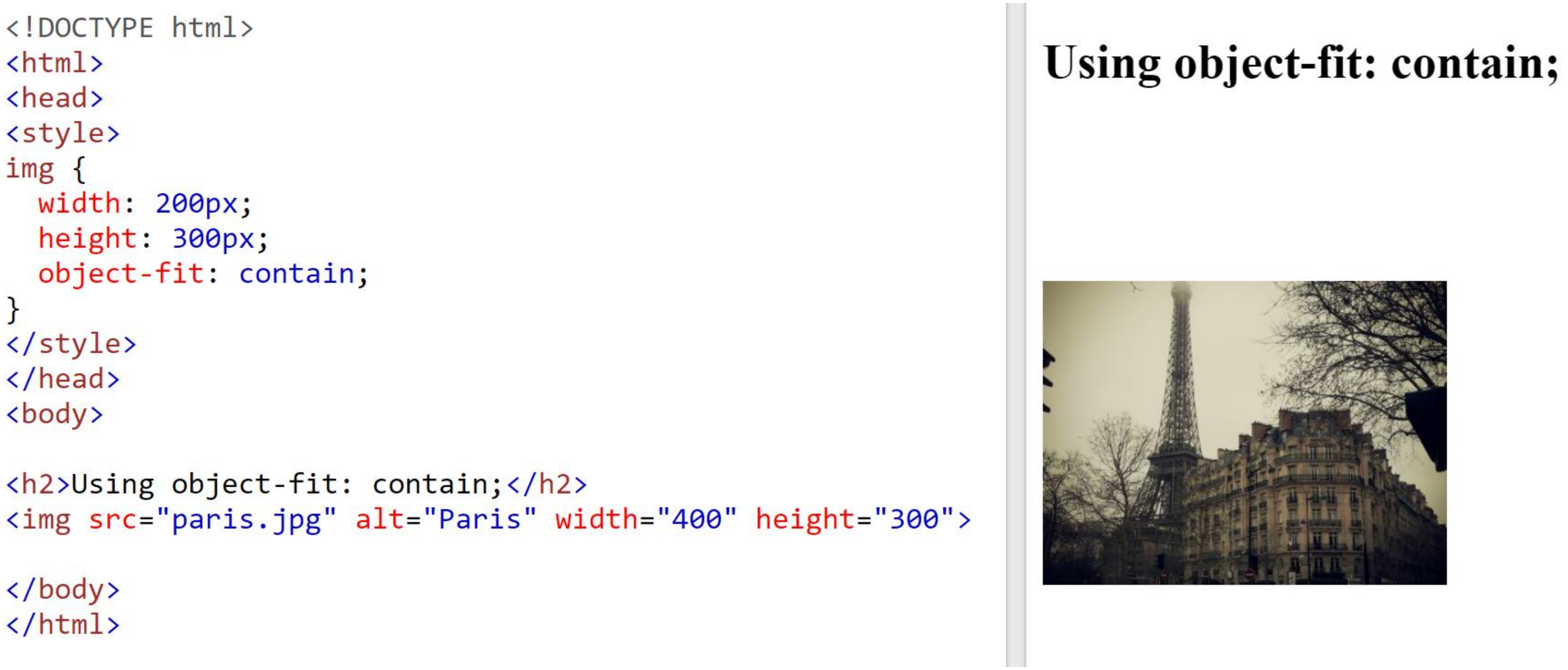
However, if we style the image above to be half its width (200 pixels) and same height (300 pixels), it will look like this:





```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  width: 200px;
  height: 300px;
  object-fit: cover;
}
</style>
</head>
<body>

<h2>Using object-fit: cover;</h2>
<img src="paris.jpg" alt="Paris" width="400" height="300">

</body>
</html>
```

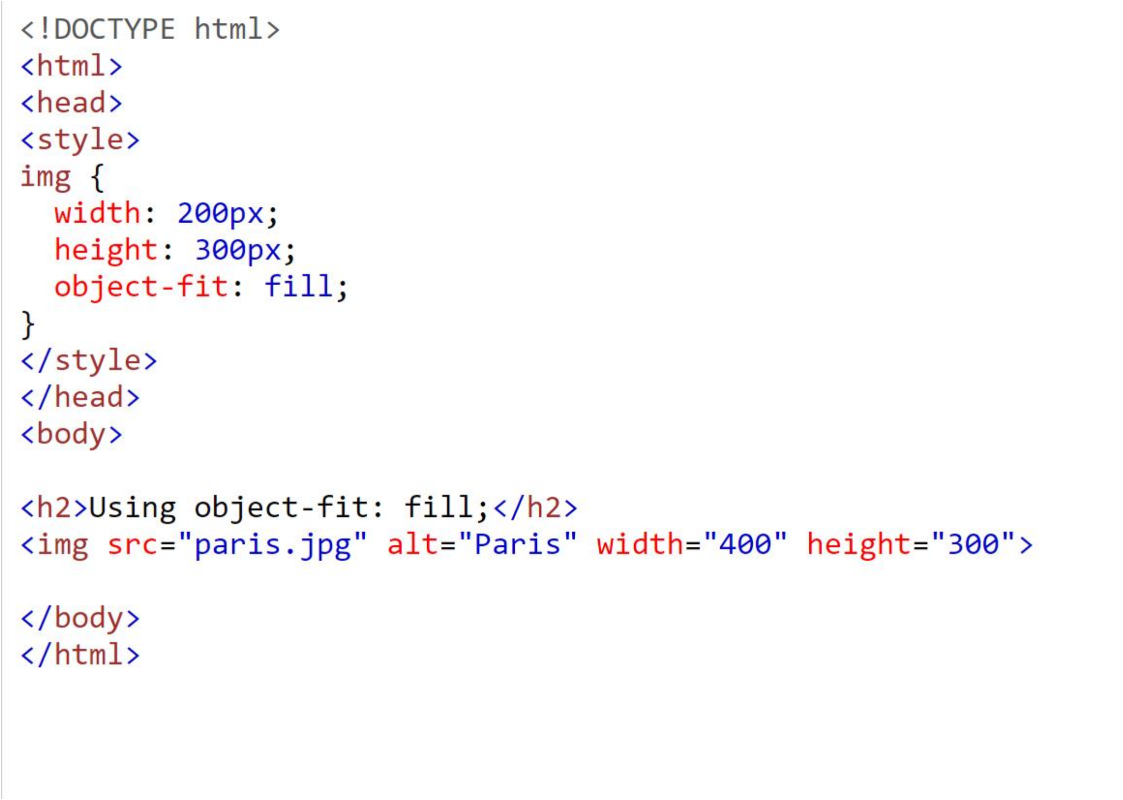
Using object-fit: cover;



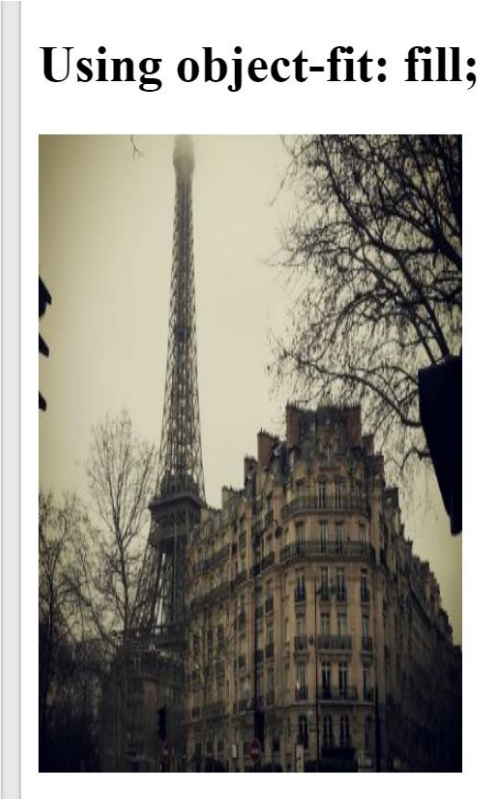
```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  width: 200px;
  height: 300px;
  object-fit: contain;
}
</style>
</head>
<body>

<h2>Using object-fit: contain;</h2>
<img src="paris.jpg" alt="Paris" width="400" height="300">

</body>
</html>
```

Using object-fit: contain;

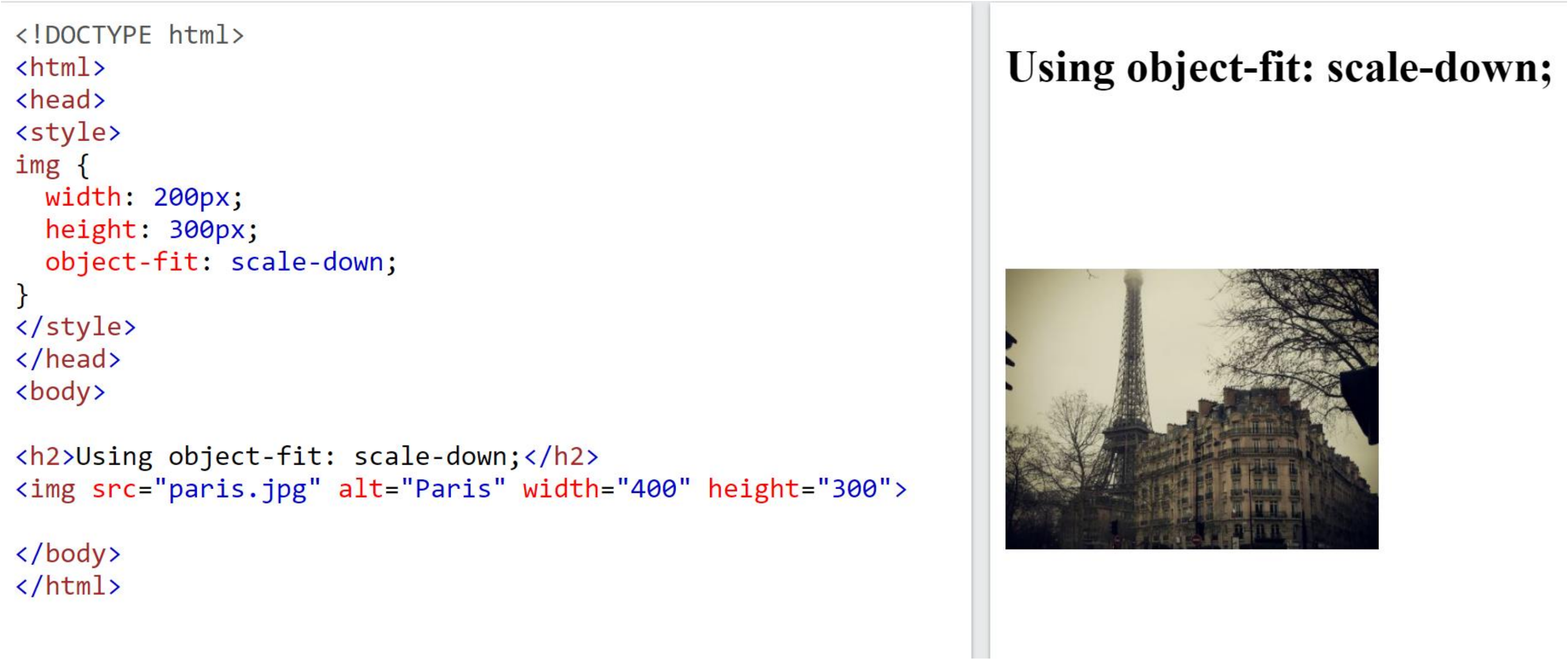
Streched-image



```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  width: 200px;
  height: 300px;
  object-fit: fill;
}
</style>
</head>
<body>

<h2>Using object-fit: fill;</h2>
<img src="paris.jpg" alt="Paris" width="400" height="300">

</body>
</html>
```



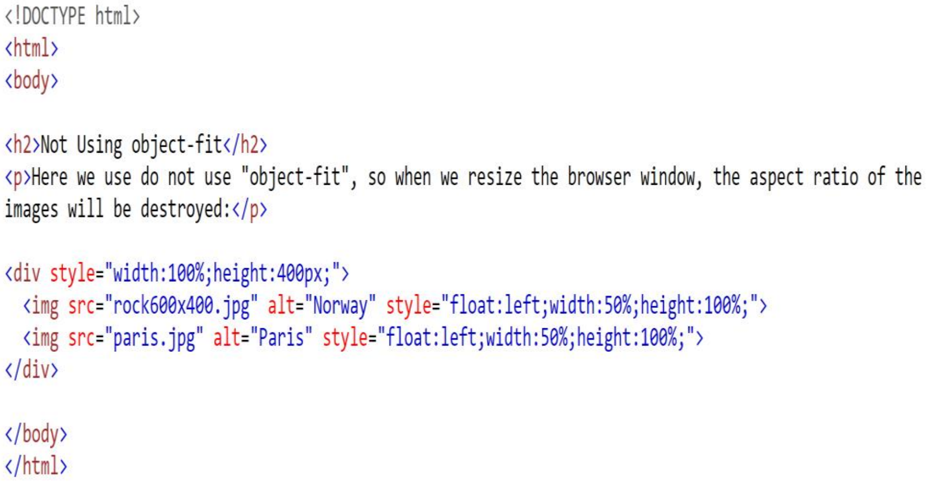
Using object-fit: fill;



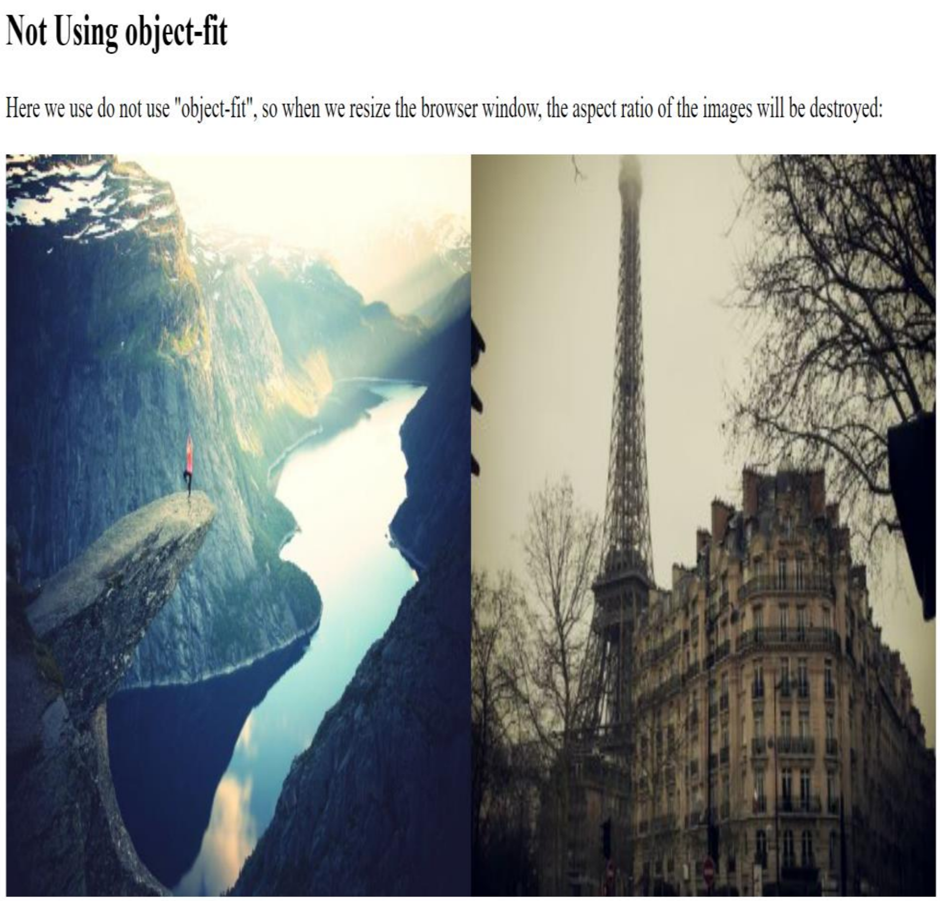
```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  width: 200px;
  height: 300px;
  object-fit: scale-down;
}
</style>
</head>
<body>

<h2>Using object-fit: scale-down;</h2>
<img src="paris.jpg" alt="Paris" width="400" height="300">

</body>
</html>
```

Using object-fit: scale-down;

Part-1



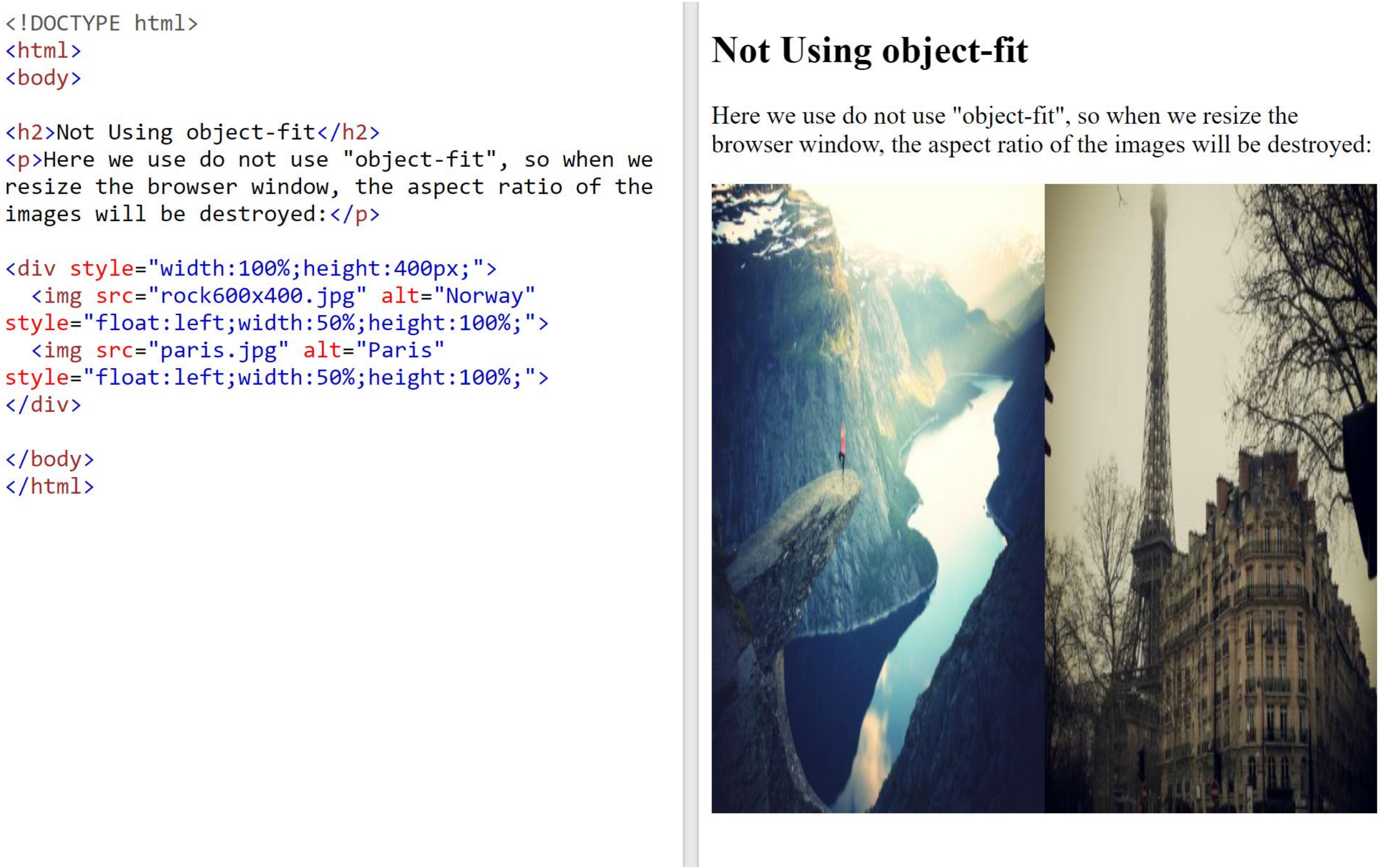
```
<!DOCTYPE html>
<html>
<body>

<h2>Not Using object-fit</h2>
<p>Here we use do not use "object-fit", so when we resize the browser window, the aspect ratio of the
images will be destroyed:</p>

<div style="width:100%;height:400px;">
  <img src="rock600x400.jpg" alt="Norway" style="float:left;width:50%;height:100%;">
  <img src="paris.jpg" alt="Paris" style="float:left;width:50%;height:100%;">
</div>

</body>
</html>
```



Not Using object-fit

Here we use do not use "object-fit", so when we resize the browser window, the aspect ratio of the images will be destroyed:

Part2

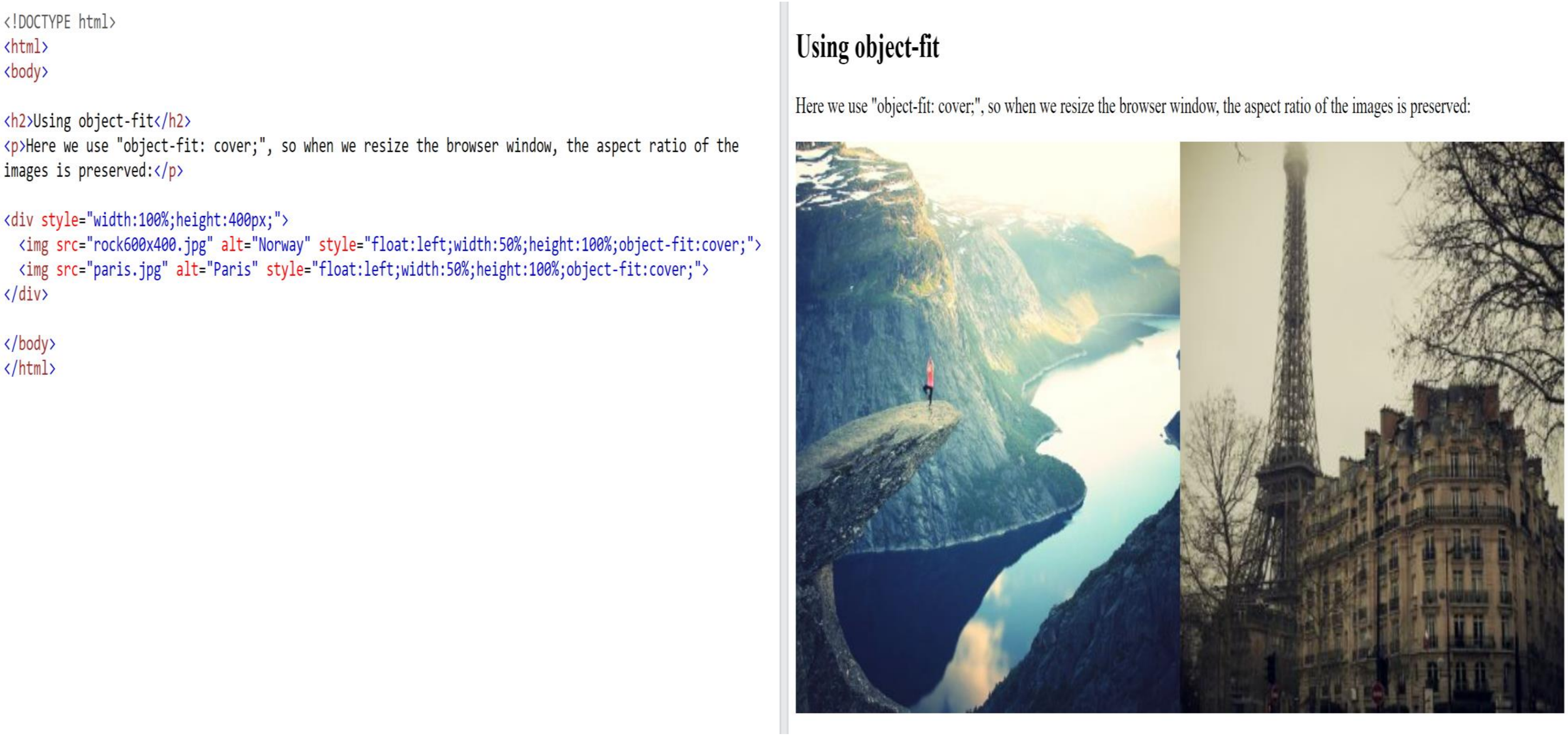


```
<!DOCTYPE html>
<html>
<body>

<h2>Not Using object-fit</h2>
<p>Here we use do not use "object-fit", so when we
resize the browser window, the aspect ratio of the
images will be destroyed:</p>

<div style="width:100%;height:400px;">
  <img src="rock600x400.jpg" alt="Norway"
style="float:left;width:50%;height:100%;">
  <img src="paris.jpg" alt="Paris"
style="float:left;width:50%;height:100%;">
</div>

</body>
</html>
```

Not Using object-fit

Here we use do not use "object-fit", so when we resize the browser window, the aspect ratio of the images will be destroyed:

Part1

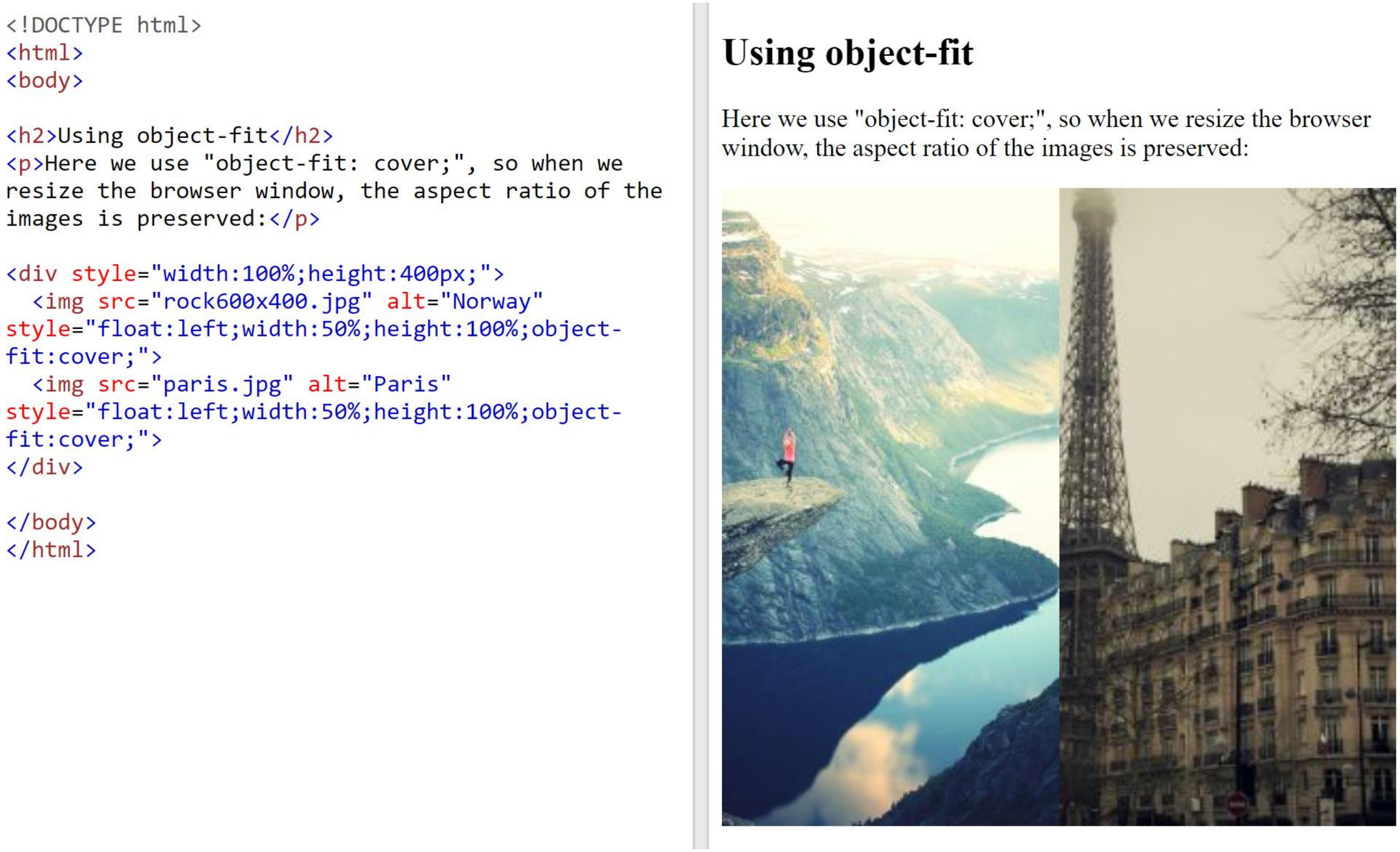


```
<!DOCTYPE html>
<html>
<body>

<h2>Using object-fit</h2>
<p>Here we use "object-fit: cover;", so when we resize the browser window, the aspect ratio of the
images is preserved:</p>

<div style="width:100%;height:400px;">
  <img src="rock600x400.jpg" alt="Norway" style="float:left;width:50%;height:100%;object-fit:cover;">
  <img src="paris.jpg" alt="Paris" style="float:left;width:50%;height:100%;object-fit:cover;">
</div>

</body>
</html>
```

Using object-fit

Here we use "object-fit: cover;", so when we resize the browser window, the aspect ratio of the images is preserved:

Part2

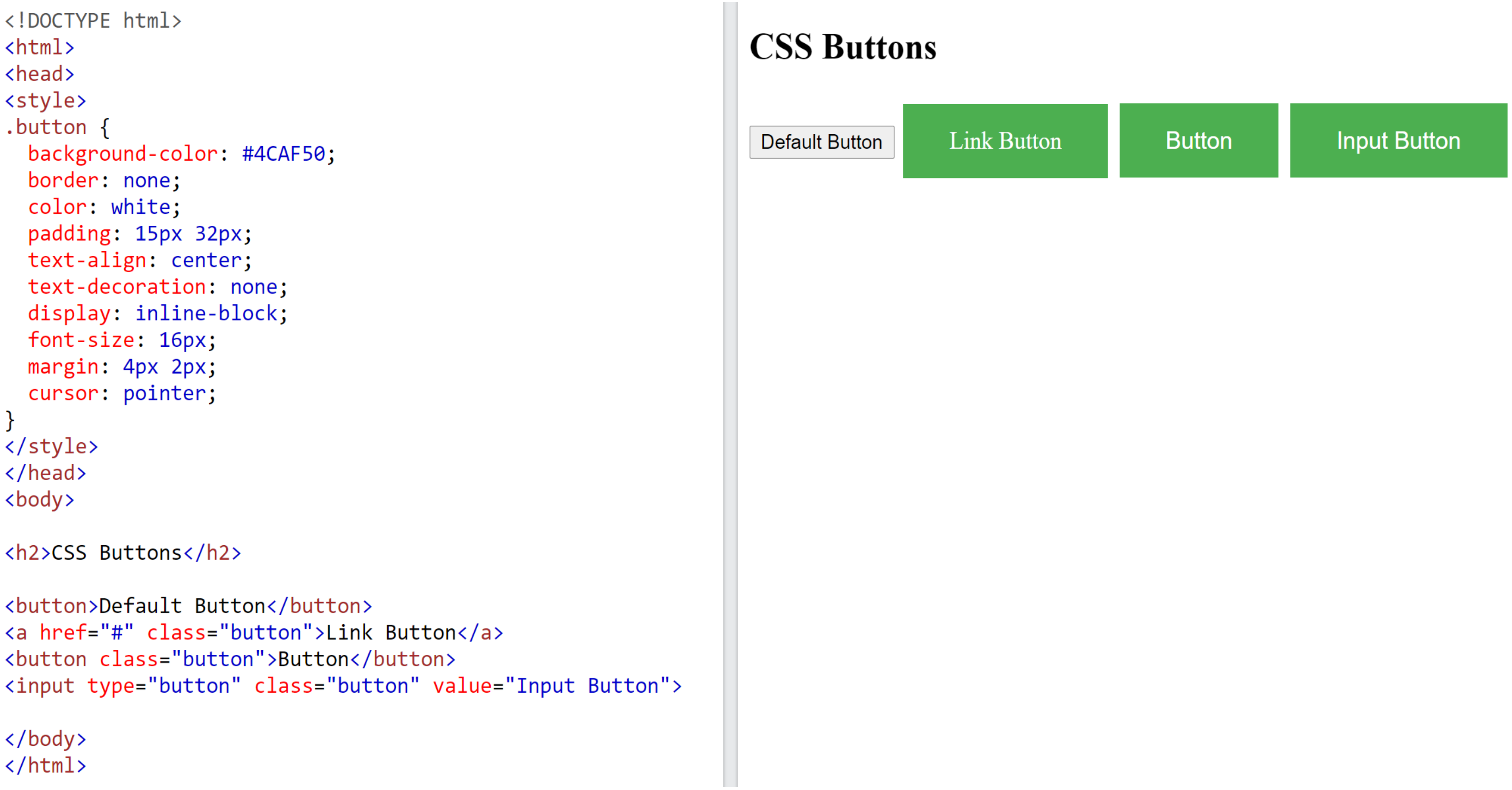


```
<!DOCTYPE html>
<html>
<body>

<h2>Using object-fit</h2>
<p>Here we use "object-fit: cover;", so when we
resize the browser window, the aspect ratio of the
images is preserved:</p>

<div style="width:100%;height:400px;">
  <img src="rock600x400.jpg" alt="Norway"
style="float:left;width:50%;height:100%;object-
fit:cover;">
  <img src="paris.jpg" alt="Paris"
style="float:left;width:50%;height:100%;object-
fit:cover;">
</div>

</body>
</html>
```

Using object-fit

Here we use "object-fit: cover;", so when we resize the browser window, the aspect ratio of the images is preserved:

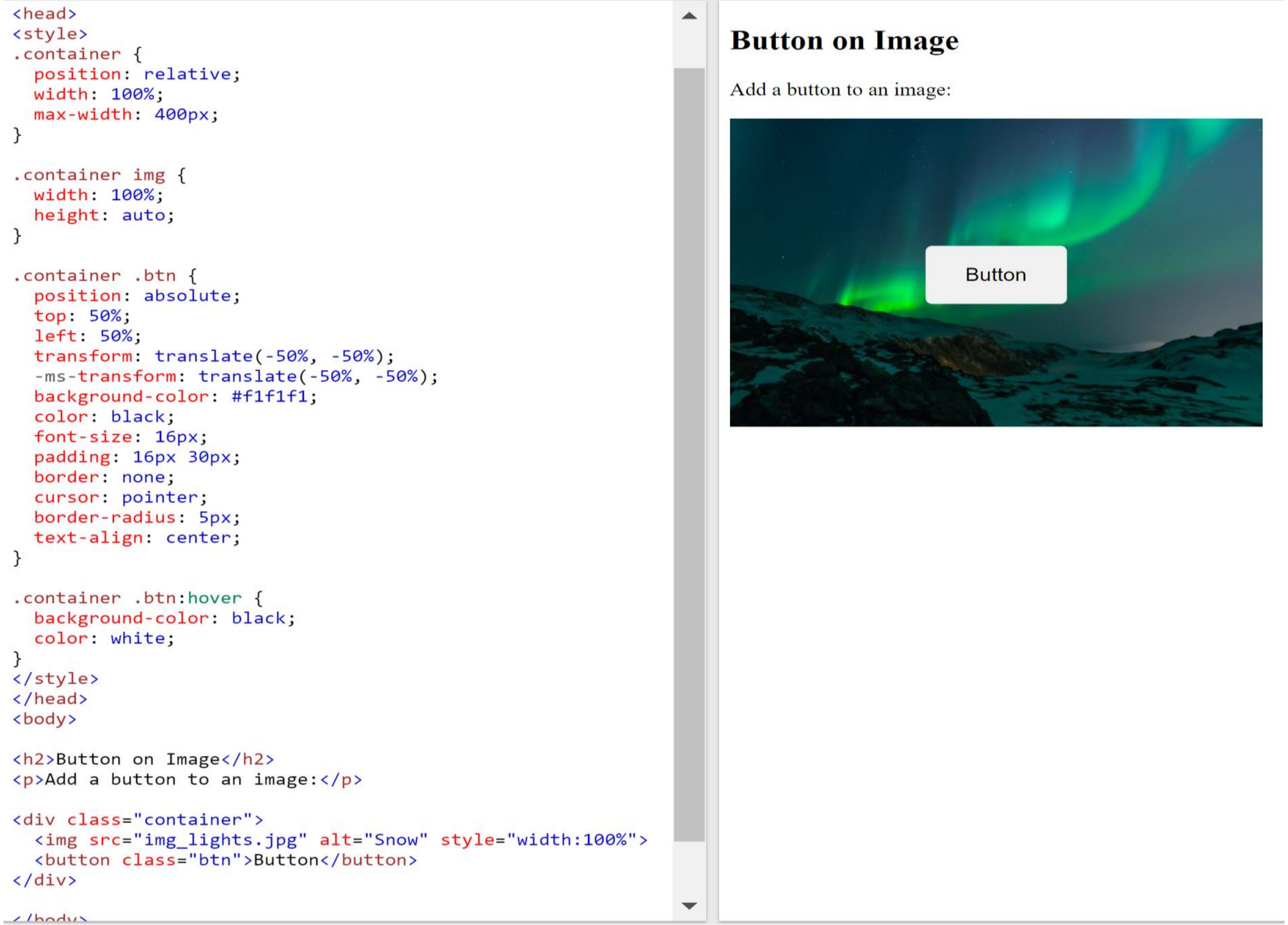
Button



```
<!DOCTYPE html>
<html>
<head>
<style>
.button {
  background-color: #4CAF50;
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
}
</style>
</head>
<body>

<h2>CSS Buttons</h2>

<button>Default Button</button>
<a href="#" class="button">Link Button</a>
<button class="button">Button</button>
<input type="button" class="button" value="Input Button">

</body>
</html>
```

CSS Buttons

- Button Color
- Button Width
- Button size
- Use the padding property to change the padding of a button
- Rounded Button
- Colored button border
- Hoverable button
- Shadowed button
- Disabled button
- Button groups



```
<head>
<style>
.container {
  position: relative;
  width: 100%;
  max-width: 400px;
}

.container img {
  width: 100%;
  height: auto;
}

.container .btn {
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  -ms-transform: translate(-50%, -50%);
  background-color: #f1f1f1;
  color: black;
  font-size: 16px;
  padding: 16px 30px;
  border: none;
  cursor: pointer;
  border-radius: 5px;
  text-align: center;
}

.container .btn:hover {
  background-color: black;
  color: white;
}
</style>
</head>
<body>

<h2>Button on Image</h2>
<p>Add a button to an image:</p>

<div class="container">
  <img src="img_lights.jpg" alt="Snow" style="width:100%">
  <button class="btn">Button</button>
</div>

</body>
```

Button on Image

Add a button to an image:

- Animated button
- Animated button with press effect

Pagination

- Explore it yourself

- The resize property specifies if (and how) an element should be resizable by the user.
- The outline-offset property adds space between an outline and the edge or border of an element.

Responsive Web Design

- What is Responsive Web Design?
  - Responsive web design makes your web page look good on all devices.
  - Responsive web design uses only HTML and CSS.
  - Responsive web design is not a program or a JavaScript.
  - Web pages can be viewed using many different devices: desktops, tablets, and phones. Your web page should look good, and be easy to use, regardless of the device.
  - Web pages should not leave out information to fit smaller devices, but rather adapt its content to fit any device