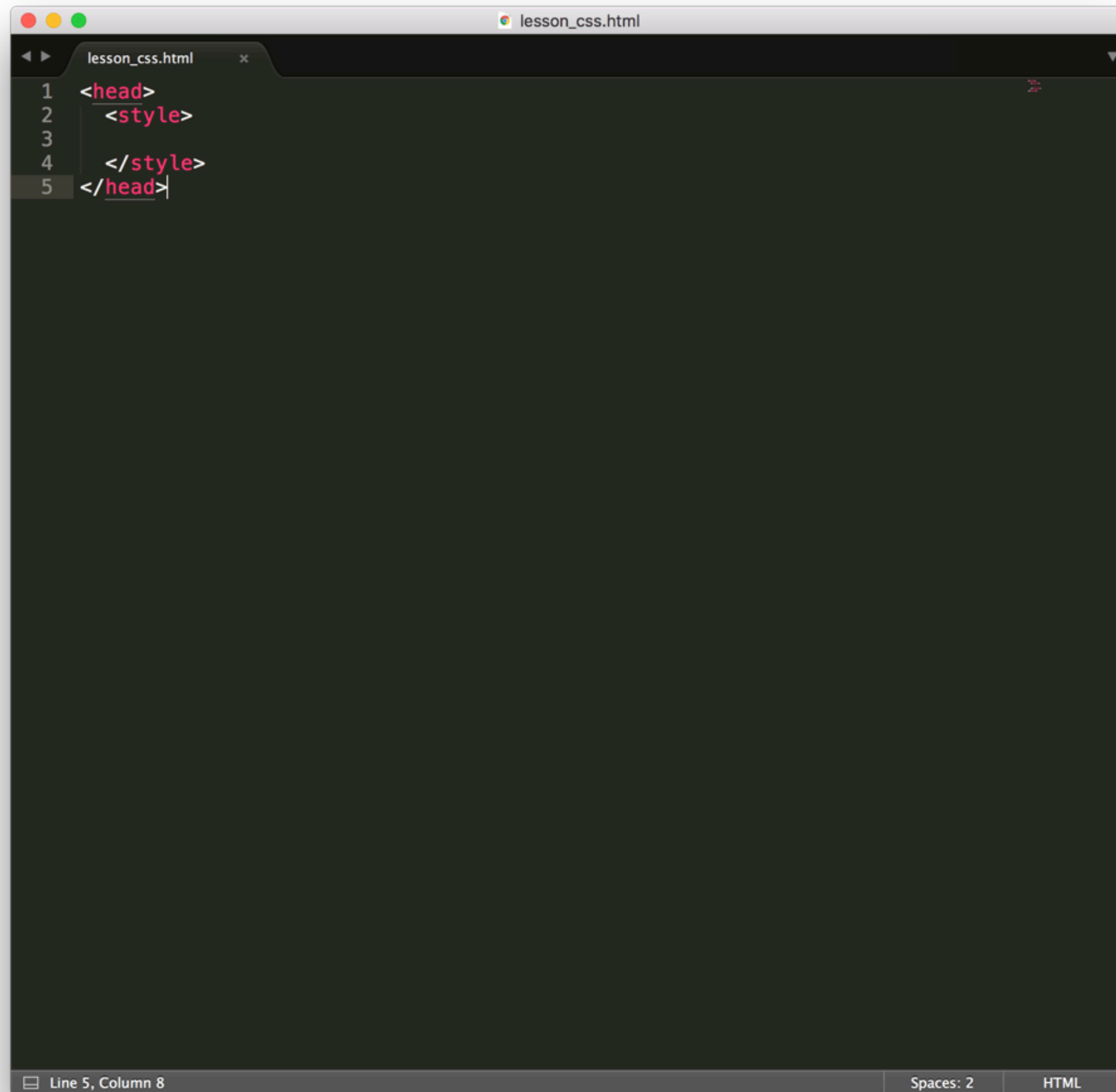


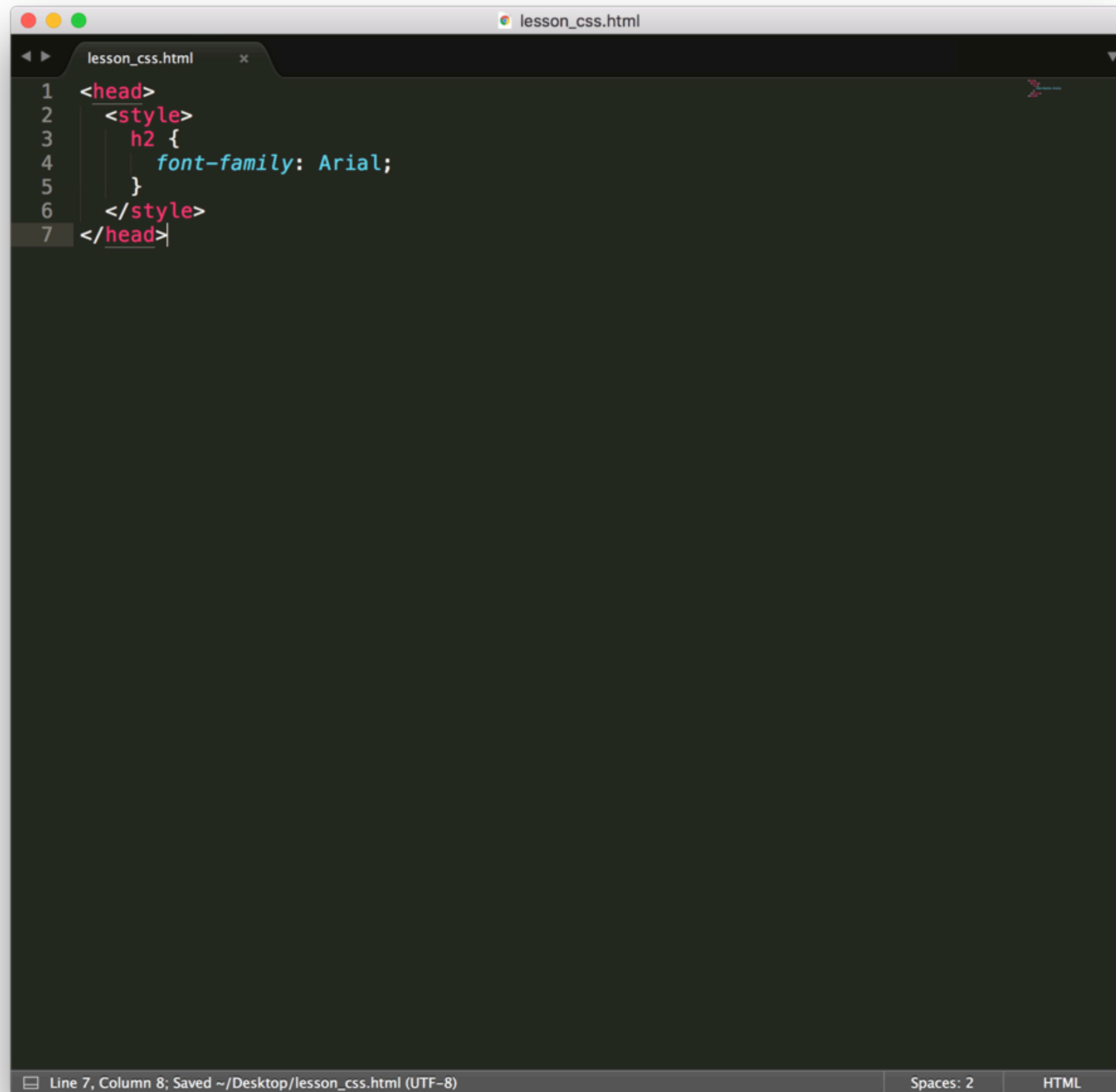
CSS

CSS, or Cascading Style Sheets, is a language that web developers use to *style* the HTML content on a web page. If you're interested in modifying colors, font types, font sizes, shadows, images, element positioning, and more, CSS is the tool to use.



```
1 <head>
2   <style>
3
4   </style>
5 </head>
```

Although CSS is a different language than HTML, it's possible to write CSS code directly within an HTML file. This is possible because of the **<style>** element.



```
1 <head>
2   <style>
3     h2 {
4       font-family: Arial;
5     }
6   </style>
7 </head>
```

The screenshot shows a code editor window titled 'lesson_css.html'. The code is written in HTML and CSS. The CSS rule defines the font-family for h2 elements as Arial. The code is as follows:

```
<head>
  <style>
    h2 {
      font-family: Arial;
    }
  </style>
</head>
```

The status bar at the bottom indicates 'Line 7, Column 8; Saved ~/Desktop/lesson_css.html (UTF-8)', 'Spaces: 2', and 'HTML'.

Once **<style>** is placed in the web page's head, we can begin writing CSS code.

See an example on the left.

Although the `<style>` element allows you to write CSS code within HTML files, this mixture of HTML and CSS can result in code that is difficult to read and maintain.

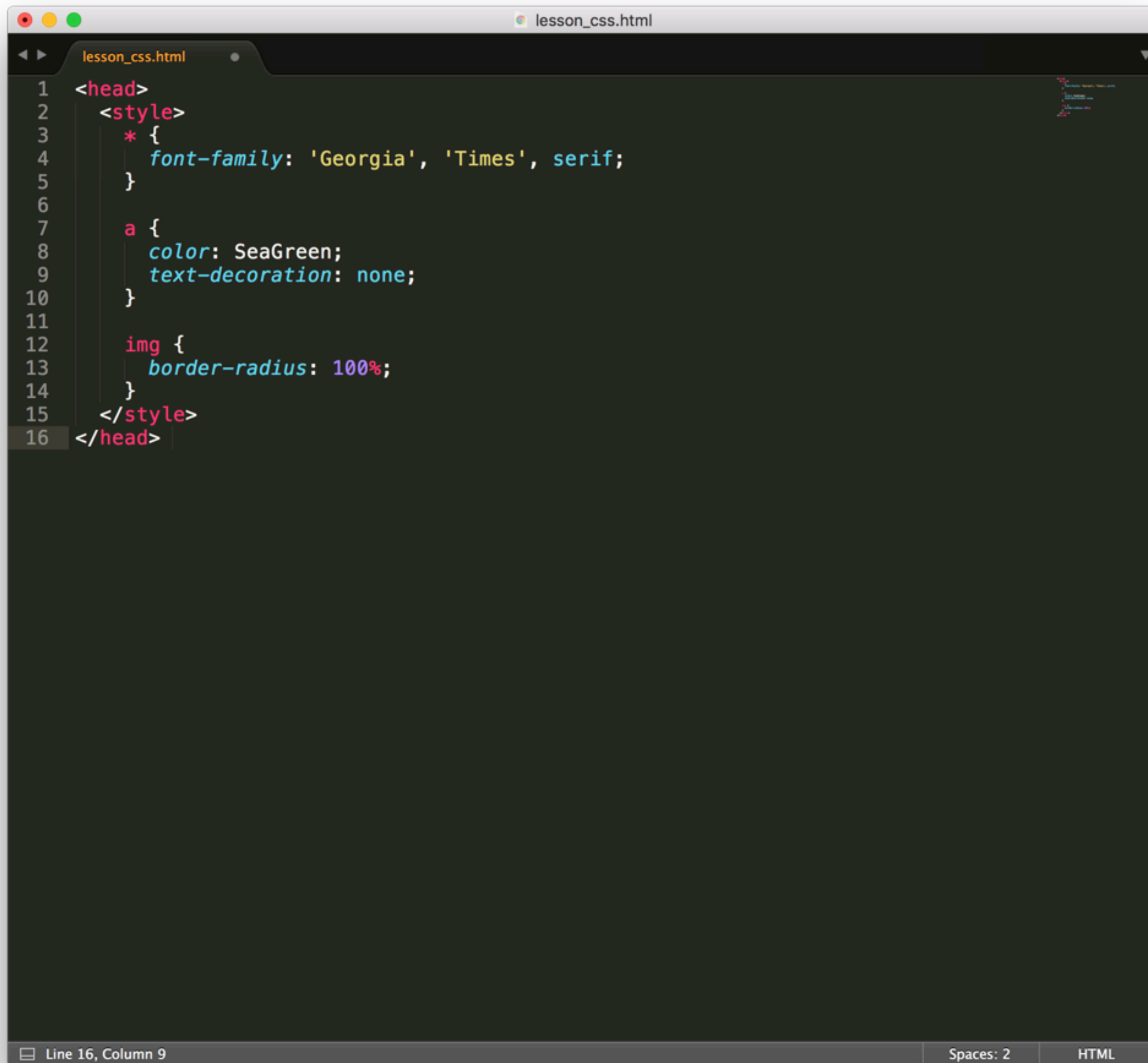
It's common for developers to add substantial amounts of custom CSS styling to a web page. When all of that CSS code is placed within a `<style>` element in an HTML file, you risk the following two things:

1. Creating a large HTML file that is difficult to read and maintain (by you and other developers). Overall, this can result in an inefficient workflow.
2. Maintaining a clear distinction between web page structure (HTML) and web page styling (CSS).

**Fortunately, the following solution will help you
avoid creating large HTML files that mix in CSS code:
a CSS file!**

HTML files are meant to contain only HTML code. Similarly, CSS files are meant to contain only CSS code. You can create a CSS file by using the .css file name extension, like so: style.css

With a CSS file, you can write all the CSS code needed to style a page without having to sacrifice the readability and maintainability of your HTML file.

A screenshot of a code editor window titled 'lesson_css.html'. The editor shows CSS code within a <head> <style> block. The code is as follows:

```
1 <head>
2   <style>
3     * {
4       font-family: 'Georgia', 'Times', serif;
5     }
6
7     a {
8       color: SeaGreen;
9       text-decoration: none;
10    }
11
12    img {
13      border-radius: 100%;
14    }
15  </style>
16 </head>
```

The editor has a dark theme. The status bar at the bottom indicates 'Line 16, Column 9', 'Spaces: 2', and 'HTML'.

Trying making an index.html file and a style.css file.

In the html file add an anchor and image. Then in your css file type the code on the left.

Be sure to only take the css code!

Notice that nothing looks any different when you refresh the page. Why?

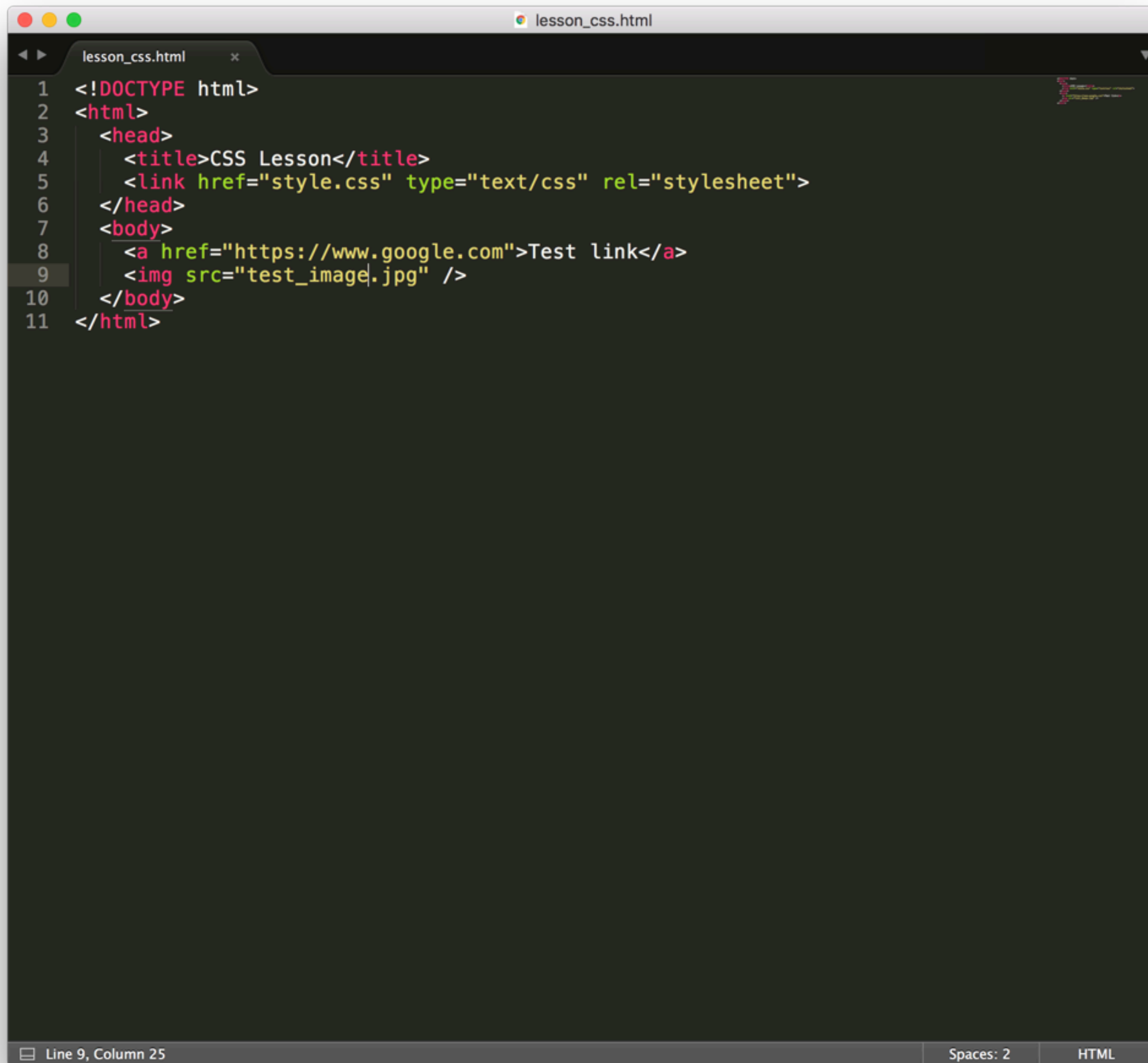
When HTML and CSS code are in separate files, the HTML file must know exactly where the CSS code is kept, otherwise, the styling can't be applied to the web page. In order to apply the styling to the web page, we'll have to *link* the HTML file and the CSS file together.

You can use the `<link>` element to link the HTML and CSS files together. The `<link>` element must be placed within the head of the HTML file. It is a self-closing tag and requires the following three attributes:

href – like the anchor element, the value of this attribute must be the address, or path, to the CSS file.

type – this attribute describes the type of document that you are linking to (in this case, a CSS file). The value of this attribute should be set to “text/css”.

rel – this attribute describes the relationship between the HTML file and the CSS file. Because you are linking to a stylesheet, the value should be set to “stylesheet”.



```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>CSS Lesson</title>
5     <link href="style.css" type="text/css" rel="stylesheet">
6   </head>
7   <body>
8     <a href="https://www.google.com">Test link</a>
9     
10  </body>
11 </html>
```

Line 9, Column 25 Spaces: 2 HTML

When linking an HTML file and a CSS file together, the **<link>** element will look like this.

Let's review what you've learned so far:

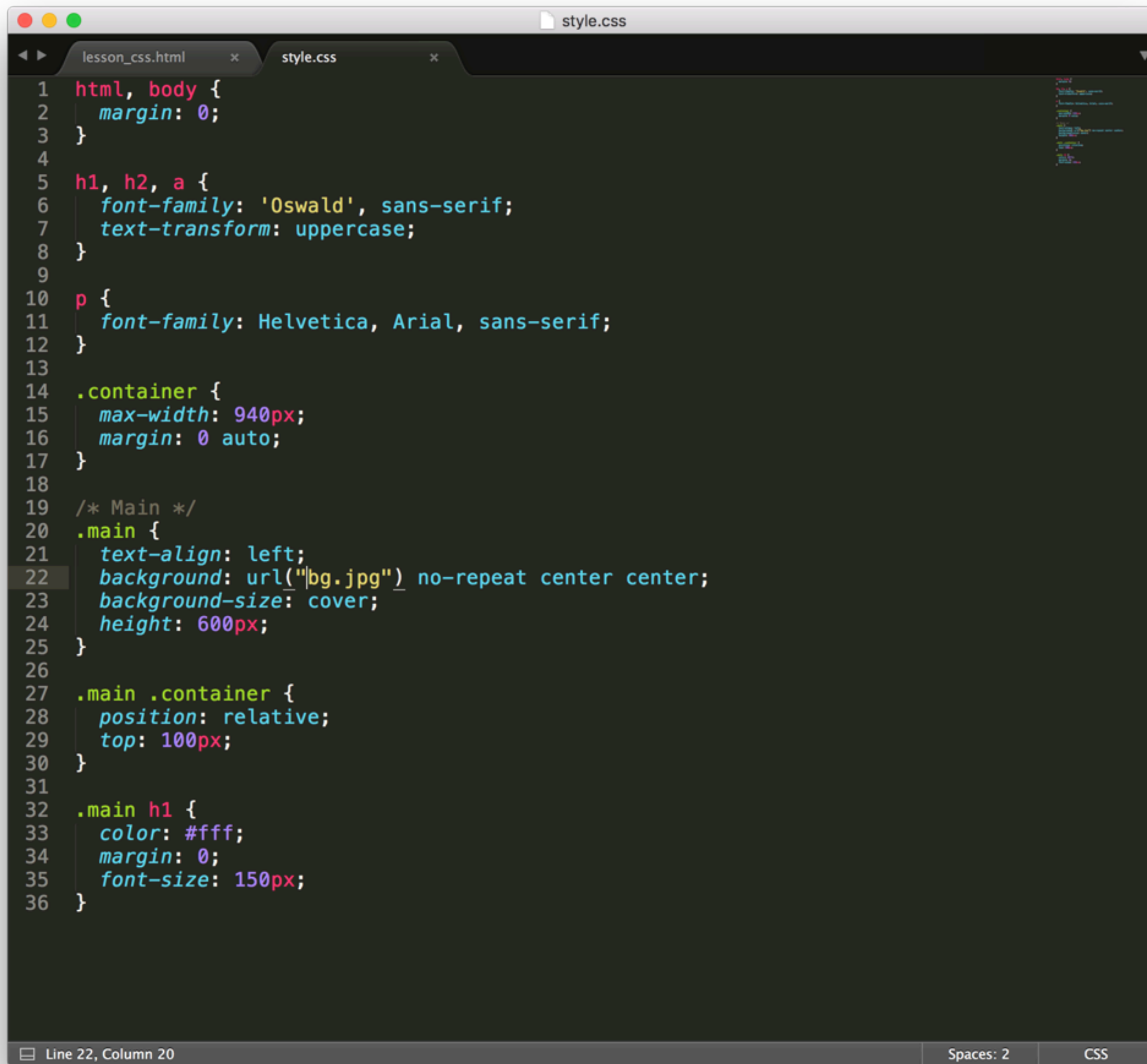
1. HTML and CSS are kept in separate files in order to keep code maintainable and readable, as well as keep structure separate from styling.
2. The `<style>` element allows you to write CSS code within an HTML file.
3. A CSS stylesheet can be linked to an HTML file using the `<link>` element, which requires three attributes:

`href` – set equal to the path of the CSS file.

`type` – set equal to “text/css”.

`rel` – set equal to “stylesheet”.

You've learned how to separate HTML and CSS into two files and link them together, but you haven't learned how CSS syntax is used to style elements on a web page.

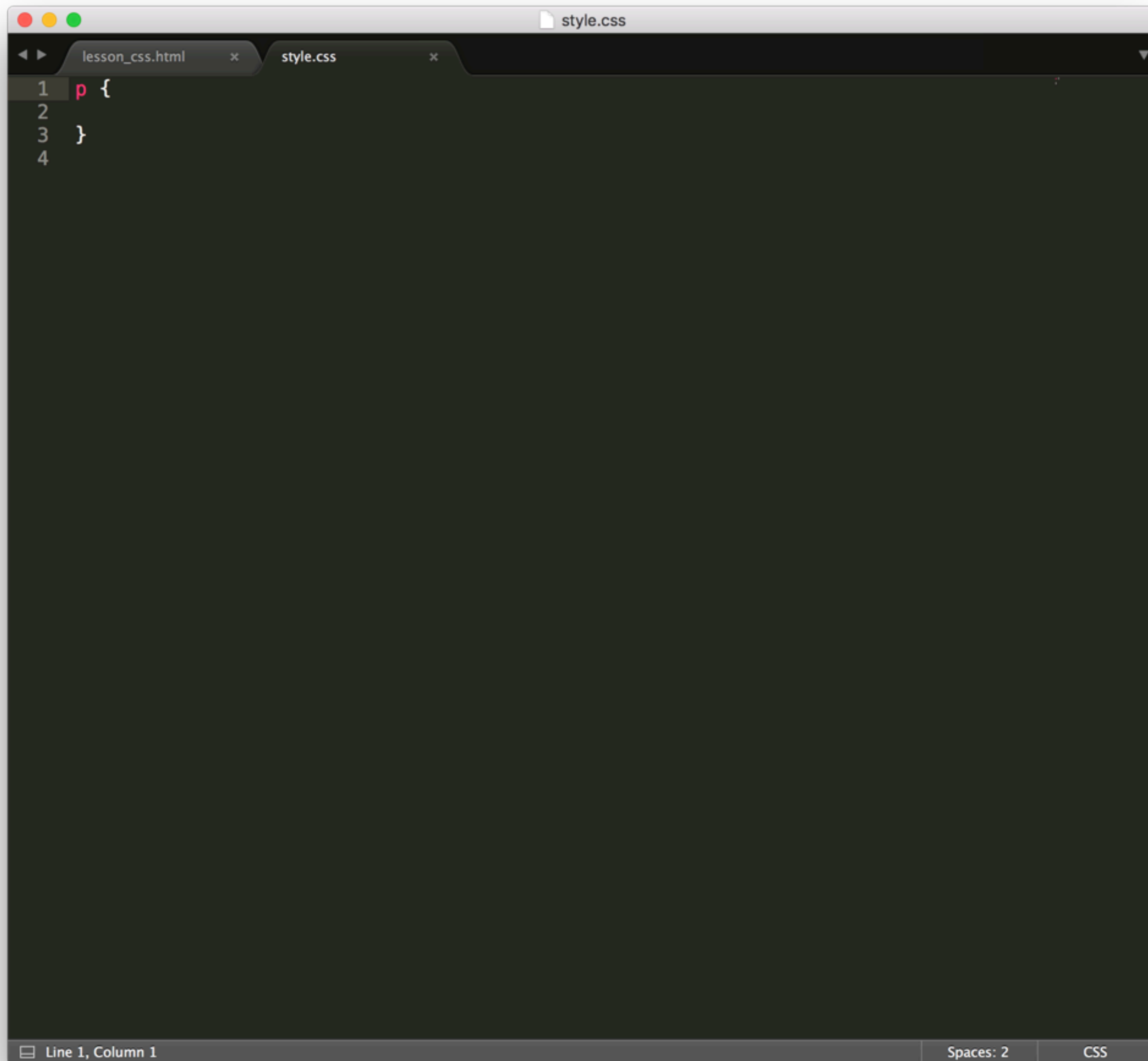


```
1  html, body {
2    margin: 0;
3  }
4
5  h1, h2, a {
6    font-family: 'Oswald', sans-serif;
7    text-transform: uppercase;
8  }
9
10 p {
11   font-family: Helvetica, Arial, sans-serif;
12 }
13
14 .container {
15   max-width: 940px;
16   margin: 0 auto;
17 }
18
19 /* Main */
20 .main {
21   text-align: left;
22   background: url("bg.jpg") no-repeat center center;
23   background-size: cover;
24   height: 600px;
25 }
26
27 .main .container {
28   position: relative;
29   top: 100px;
30 }
31
32 .main h1 {
33   color: #fff;
34   margin: 0;
35   font-size: 150px;
36 }
```

Line 22, Column 20 Spaces: 2 CSS

Take a look... this is what css looks like.

To style an HTML element using CSS, you must first *select* that element in the CSS file.



```
1 p {  
2  
3 }  
4
```

The screenshot shows a code editor window with two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, displaying a CSS rule on line 1: 'p {'. The rule is incomplete, with lines 2 and 3 showing closing braces '}' and line 4 being empty. The editor's status bar at the bottom indicates 'Line 1, Column 1', 'Spaces: 2', and 'CSS'.

In the example above, all paragraph elements are selected using a CSS *selector*. The selector (in this case) is **p**. Note that the CSS selector essentially matches the HTML tag for that element, but without the angle brackets.

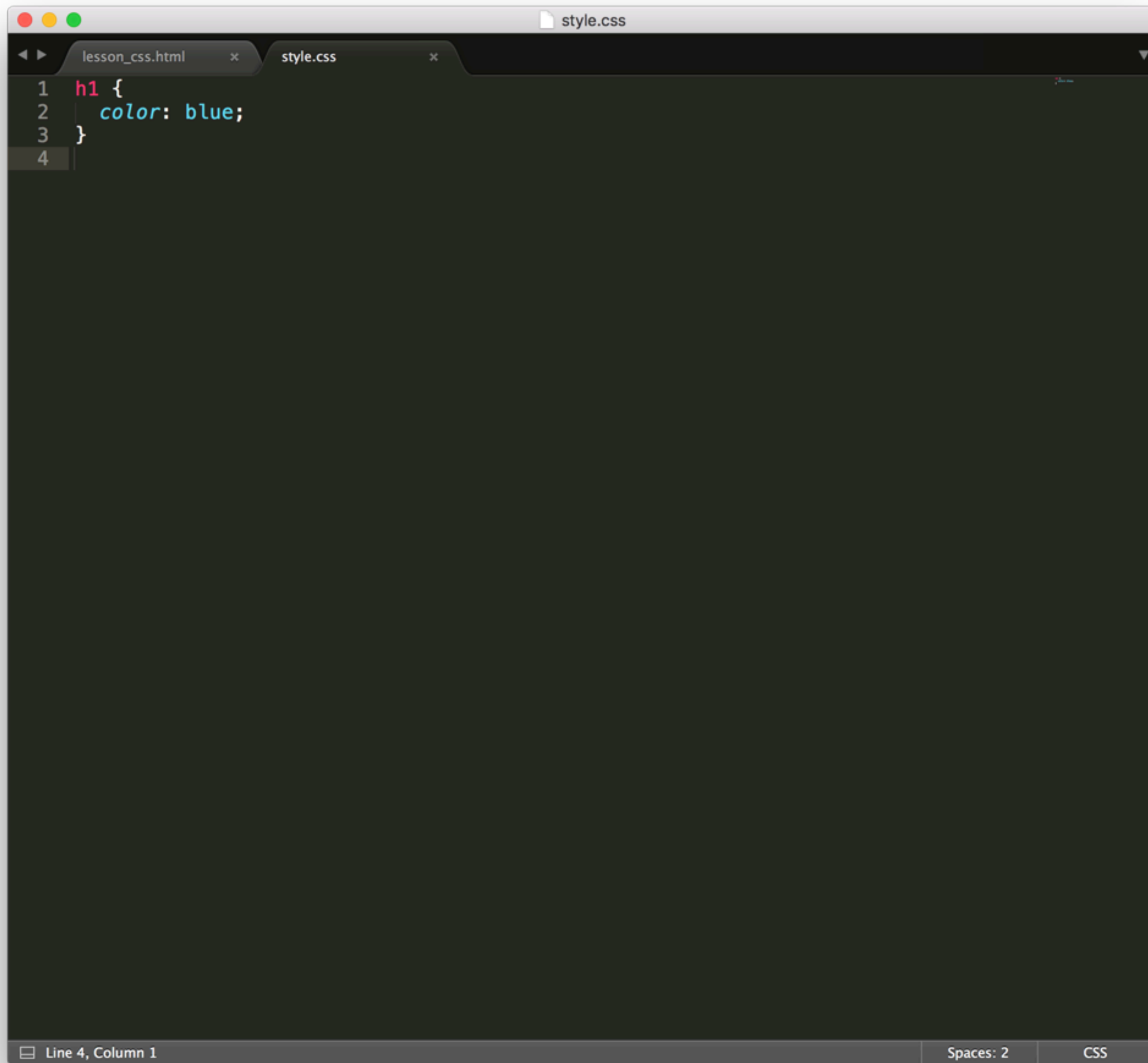
Note: The **p** selector in the example above will select *all* **<p>** elements on the web page.

It's not enough to simply select an HTML element in a CSS file. To actually style the element, you need to specify two things inside the body of the selector:

Property – the property you'd like to style of that element (i.e., size, color, etc.).

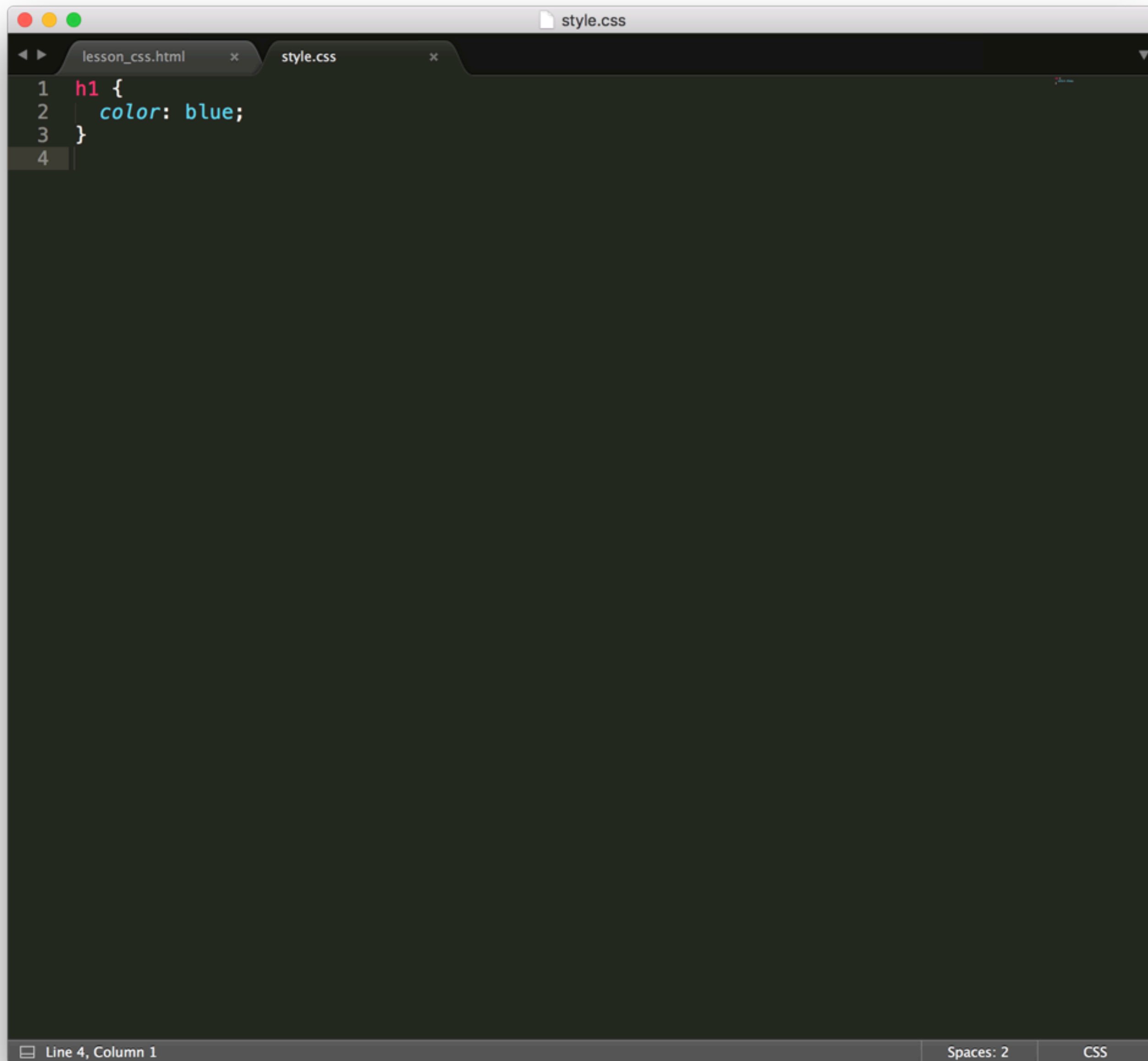
Value – the value of the property (i.e., **18px** for size, **Blue** for color, etc.).



A screenshot of a code editor window with a dark theme. The window has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing a CSS rule for the 'h1' selector. The rule is: 'h1 { color: blue; }'. The text is color-coded: 'h1' is red, '{' is blue, 'color:' is blue, 'blue;' is blue, and '}' is blue. The line numbers 1, 2, 3, and 4 are visible on the left. The status bar at the bottom shows 'Line 4, Column 1', 'Spaces: 2', and 'CSS'.

```
1 h1 {  
2   color: blue;  
3 }  
4
```

In this example, the `<h1>` element has been selected. Inside of the selector's body, the heading's `color` property is set to a value of `Blue`. The heading will now appear the color blue in the browser.

A screenshot of a code editor window. The title bar shows 'style.css'. The editor has two tabs: 'lesson_css.html' and 'style.css'. The code in the 'style.css' tab is:

```
1 h1 {  
2   color: blue;  
3 }  
4
```

The code is syntax-highlighted: 'h1' is red, '{' is blue, 'color:' is blue, 'blue;' is blue, and '}' is blue. The status bar at the bottom shows 'Line 4, Column 1', 'Spaces: 2', and 'CSS'.

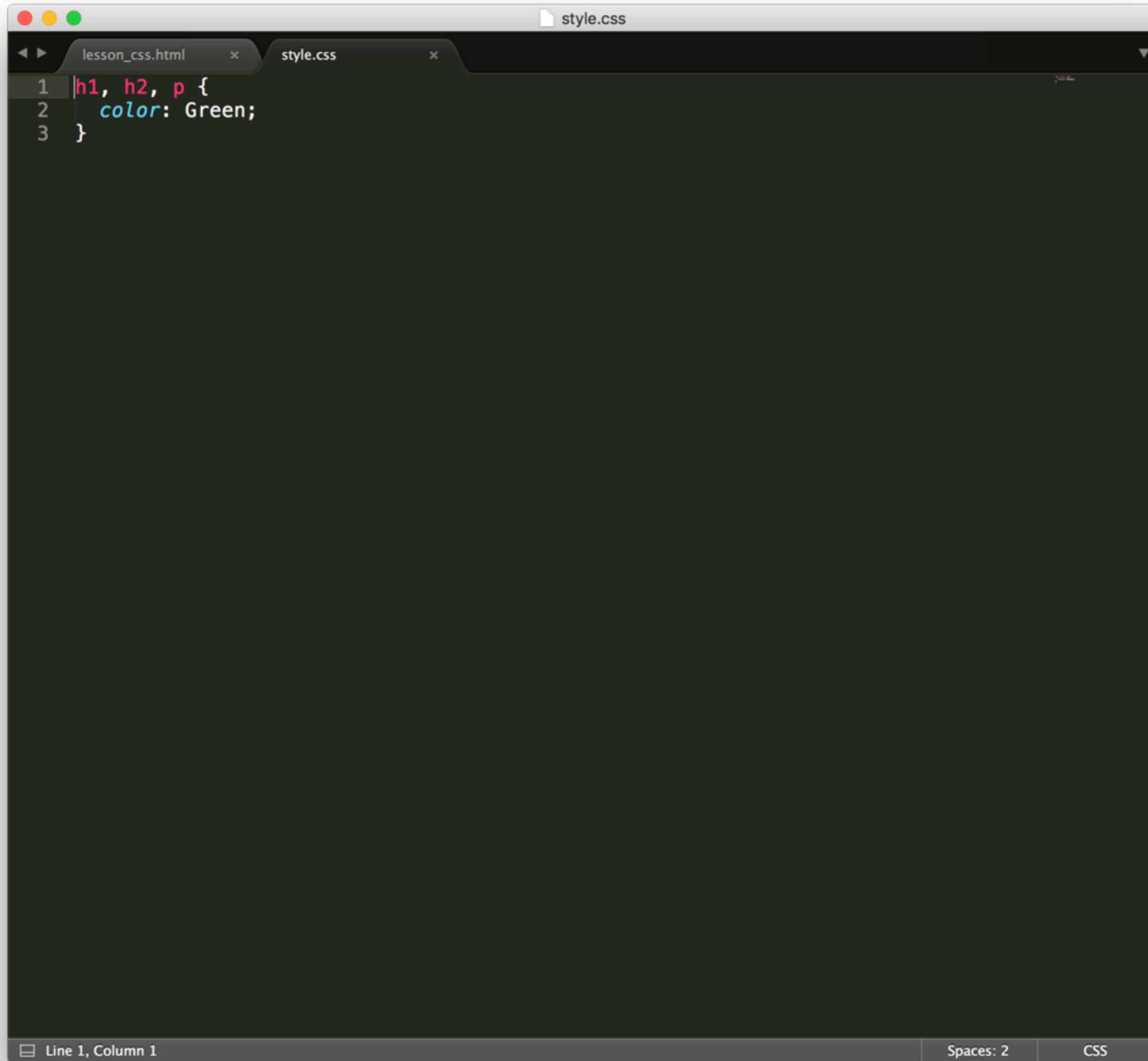
The line **color: Blue;** is referred to *CSS declaration*. A CSS declaration consists of a property and a value. Note that a semicolon (;) ends all declarations.

Finally, the entire snippet of code in this example is known as a *CSS rule*. A CSS rule consists of the selector and all declarations inside of the selector.

Let's give this a try. Add a heading tag and paragraph tag to your html document. Change the **color** of the heading to **pink** and the **font-size** of the paragraph to **35px**.

Styling with CSS would be very inefficient if you were forced to manually style the same property across many elements.

For example, what if you wanted to change the color of 10 different elements to **Aquamarine** in CSS?.



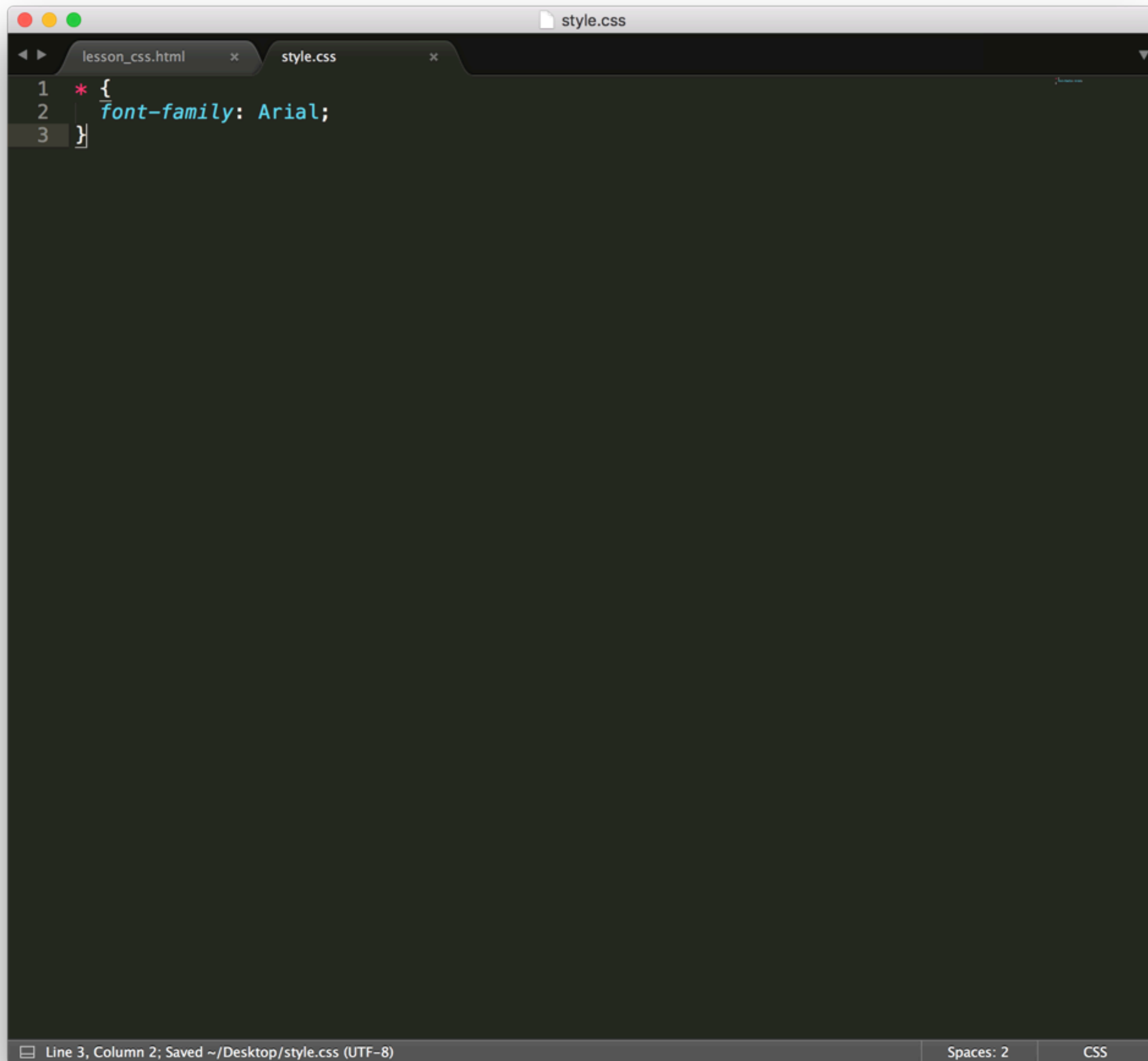
```
1 h1, h2, p {  
2   color: Green;  
3 }
```

The screenshot shows a code editor window with two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, displaying the CSS code. The code uses a multiple element selector 'h1, h2, p' to apply the 'color: Green;' property to h1, h2, and p elements. The editor has a dark theme and a status bar at the bottom showing 'Line 1, Column 1', 'Spaces: 2', and 'CSS'.

Fortunately, you can select multiple elements at once so that you can save time styling a shared property.

In this example, the `<h1>` heading, the `<h2>` heading, and the paragraph have all been styled to appear **Green** using a *multiple element selector*. A multiple element selector can save you time when you want to style the same property across many elements.

There's a special selector that can instantly select every single element on the web page: the *universal selector*.



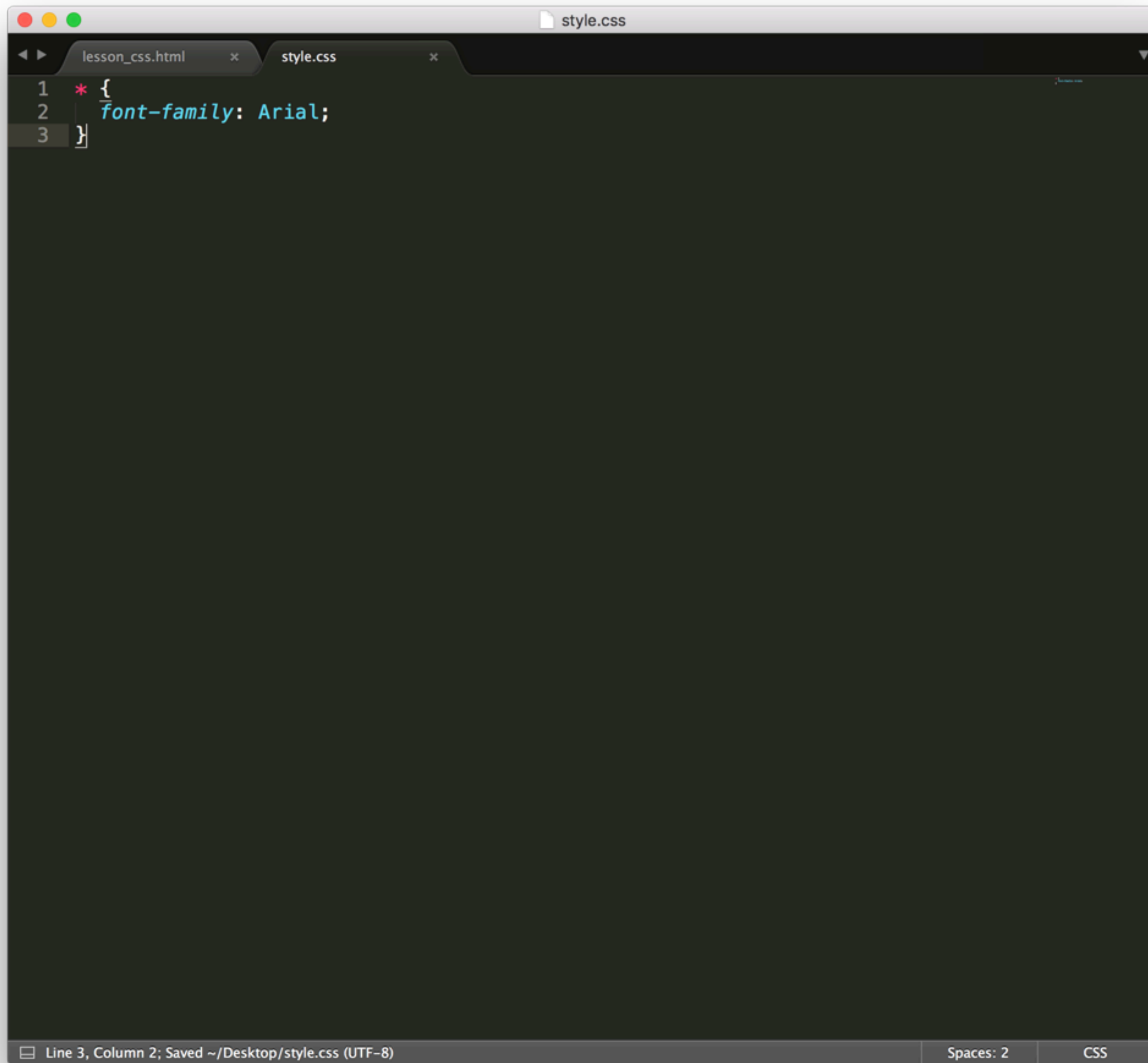
```
1 * {  
2   font-family: Arial;  
3 }
```

The screenshot shows a code editor window with two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, displaying the CSS code. The code consists of a single rule using the universal selector '*' to set the font-family to 'Arial'. The editor has a dark theme and shows line numbers on the left. The status bar at the bottom indicates 'Line 3, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

The universal selector, *****, is used to select every element on the page and set the font to **Arial**.

What makes the universal selector so special? When all elements on a web page require the same styling, it's often more efficient to set that styling using the universal selector.

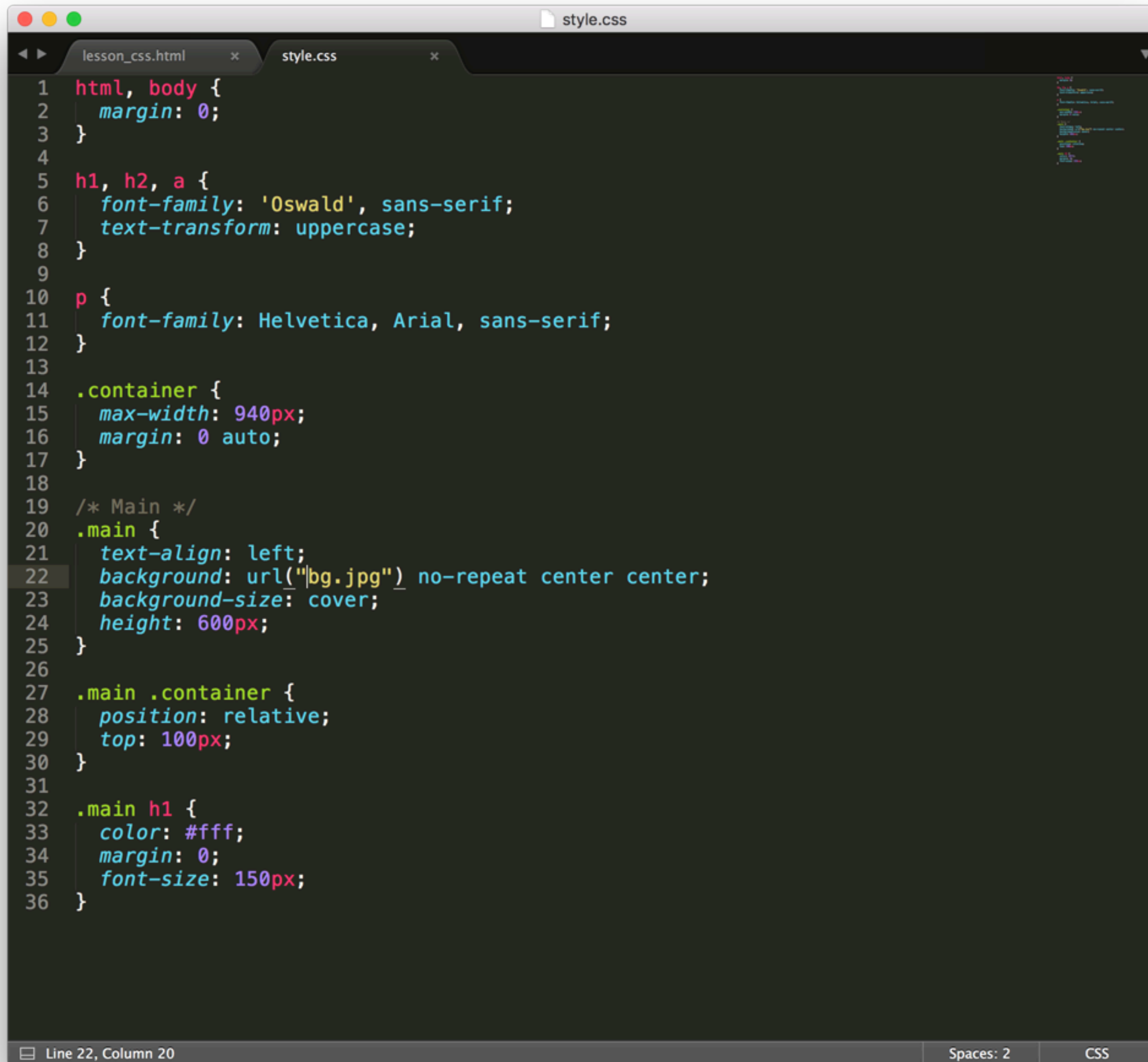
Afterwards, you can modify (or remove) that styling for specific elements that don't require it.



A screenshot of a code editor window with a dark theme. The window has a title bar with three colored buttons (red, yellow, green) and a tab labeled 'style.css'. The editor shows three lines of CSS code: line 1 has a red asterisk followed by an opening curly brace '{'; line 2 has 'font-family: Arial;' with a blue squiggly line under 'font-family'; line 3 has a closing curly brace '}'. The status bar at the bottom shows 'Line 3, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

```
1 * {  
2   font-family: Arial;  
3 }
```

Just like HTML, CSS follows certain best practices for spacing and indentation.



```
1  html, body {
2    margin: 0;
3  }
4
5  h1, h2, a {
6    font-family: 'Oswald', sans-serif;
7    text-transform: uppercase;
8  }
9
10 p {
11   font-family: Helvetica, Arial, sans-serif;
12 }
13
14 .container {
15   max-width: 940px;
16   margin: 0 auto;
17 }
18
19 /* Main */
20 .main {
21   text-align: left;
22   background: url("bg.jpg") no-repeat center center;
23   background-size: cover;
24   height: 600px;
25 }
26
27 .main .container {
28   position: relative;
29   top: 100px;
30 }
31
32 .main h1 {
33   color: #fff;
34   margin: 0;
35   font-size: 150px;
36 }
```

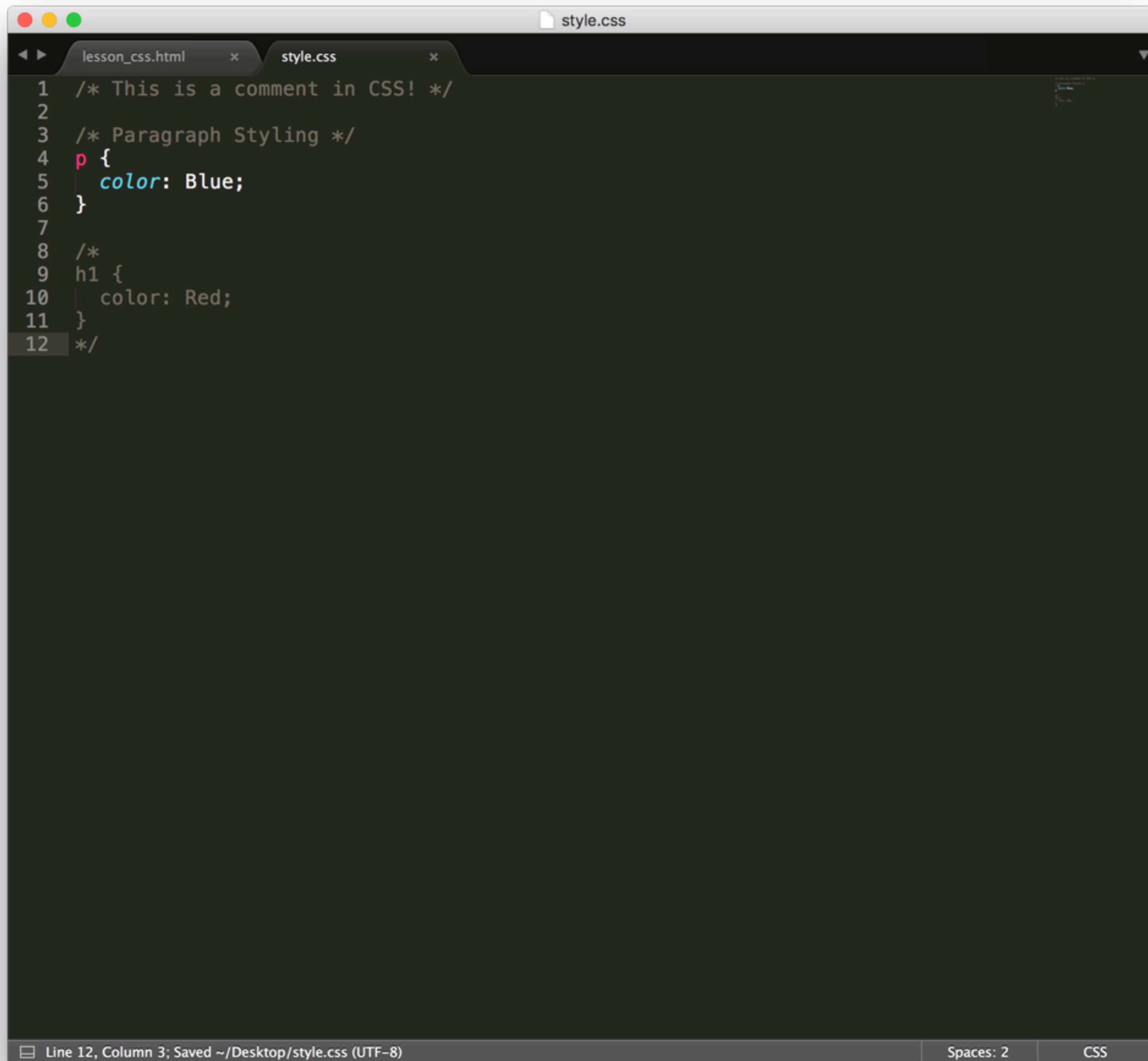
Line 22, Column 20 Spaces: 2 CSS

One space should be used between the selector and the opening curly brace (`{`).

No extra spaces should exist between opening and closing curly braces (`{` and `}`) and CSS declarations.

Two spaces of indentation should be used for CSS declarations.

One line of spacing should exist between CSS rules.

A screenshot of a code editor window with two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing the following CSS code:

```
1  /* This is a comment in CSS! */
2
3  /* Paragraph Styling */
4  p {
5      color: Blue;
6  }
7
8  /*
9  h1 {
10     color: Red;
11  }
12 */
```

The code is syntax-highlighted, with comments in grey, selectors in red, and property values in blue. The status bar at the bottom indicates 'Line 12, Column 3; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

Just like HTML, you can also leave comments in your CSS file. CSS comments begin with `/*` and end with `*/`. Including comments in your code is helpful for many reasons:

_They help you (and others) understand your code if you decide to come back and review it at a much later date.

_They allow you to experiment with new code, without having to delete old code.

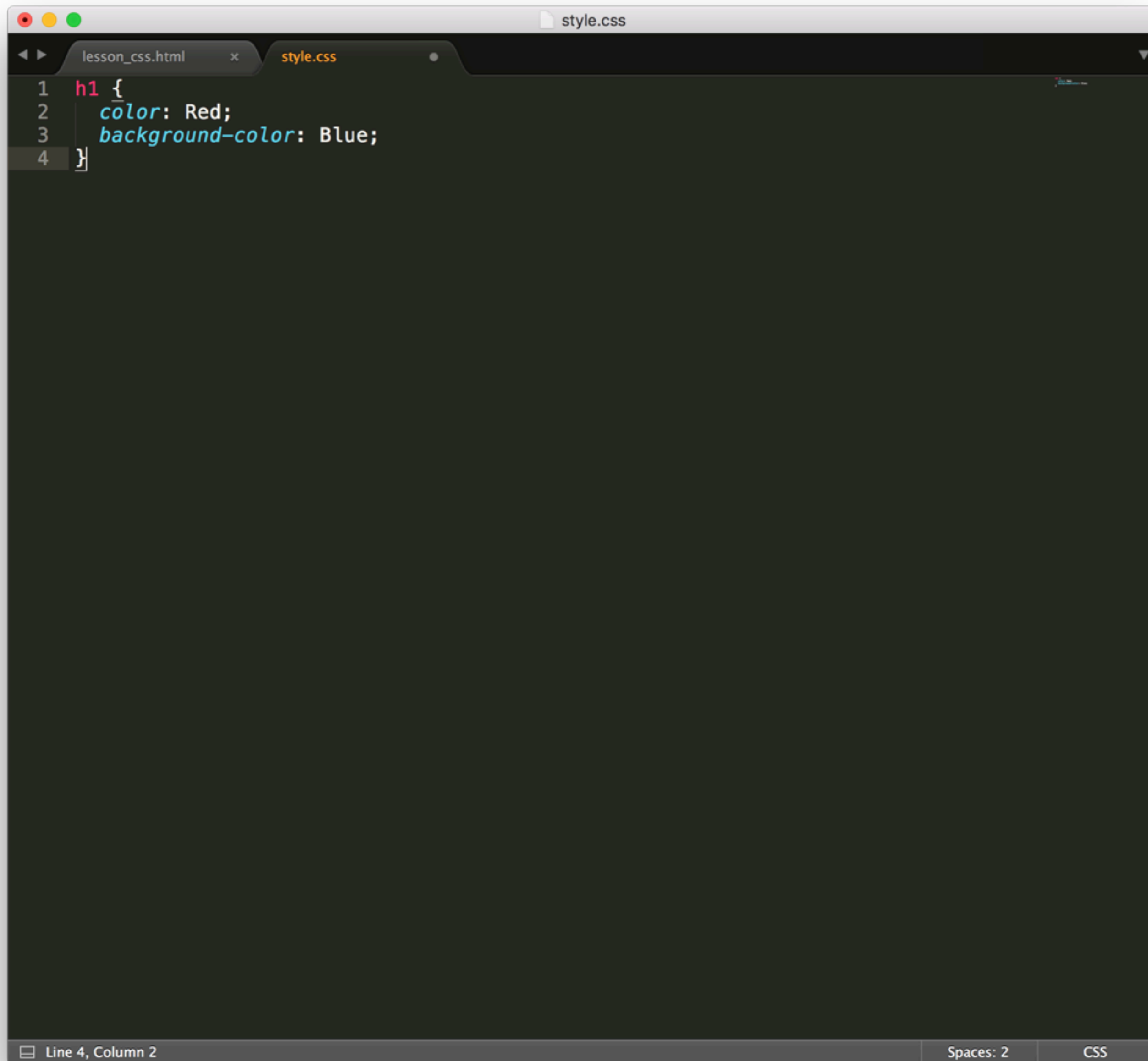
So far we've styled different properties of HTML elements, but the details of each property weren't explained. Let's start by taking a look at color.

Before discussing the specifics of color, it's important to make two distinctions about color. Color can affect the following design aspects:

The foreground color
The background color

Foreground color is the color that an element appears in. For example, when a heading is styled to appear green, the *foreground color* of the heading has been styled.

Conversely, when a heading is styled so that its background appears yellow, the *background color* of the heading has been styled



A screenshot of a code editor window with a dark theme. The window has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing the following CSS code:

```
1 h1 {  
2   color: Red;  
3   background-color: Blue;  
4 }
```

The status bar at the bottom indicates 'Line 4, Column 2', 'Spaces: 2', and 'CSS'.

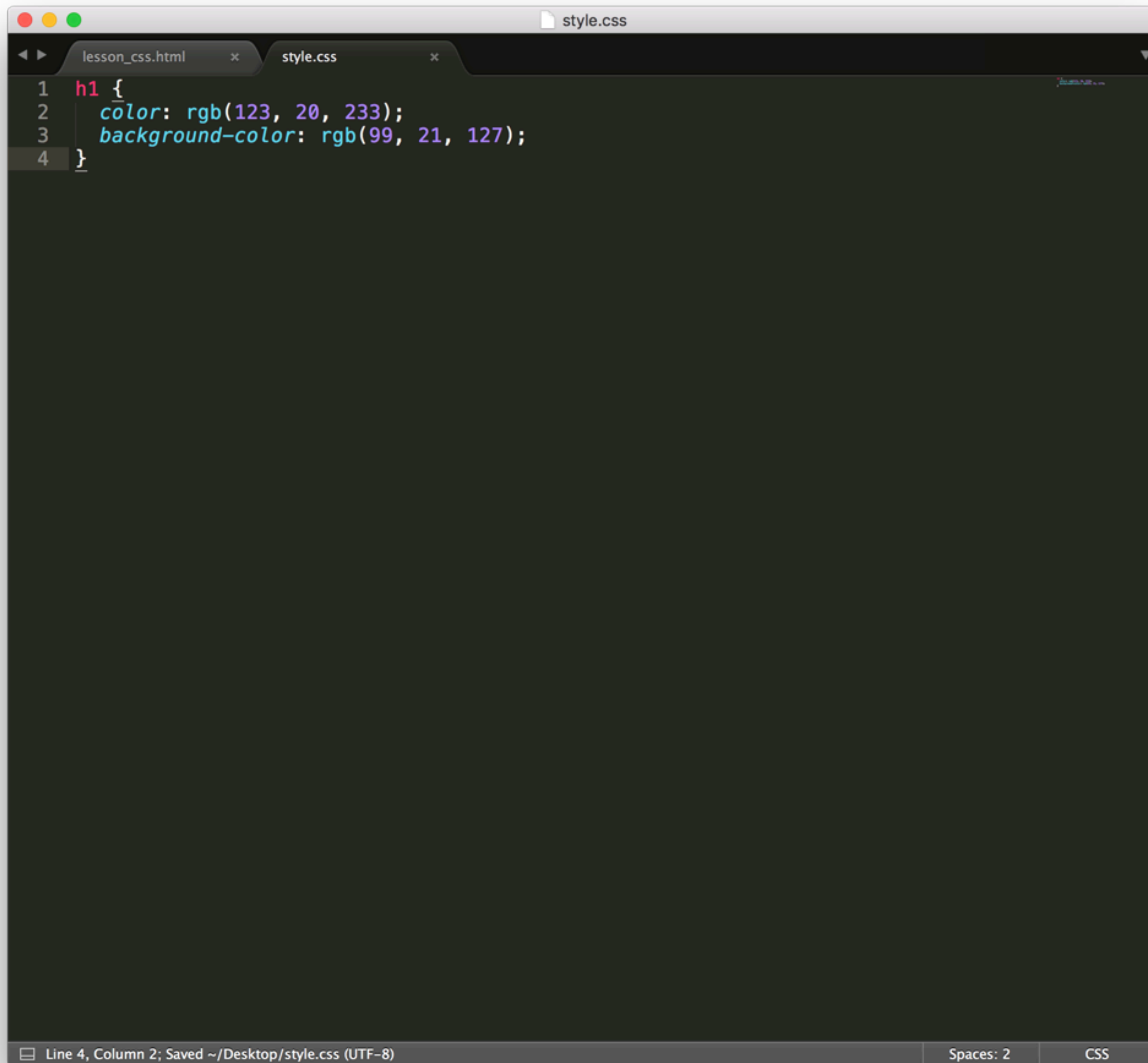
In CSS, these two design aspects can be styled with the following two properties:

color – this property styles an element's foreground color.

background-color – this property styles an element's background color.

Over the past few exercises, you've seen CSS examples that use colors like **Red**, **Blue**, or **Cyan**. In CSS, these colors are technically known as *named colors*. There are a total of **147 named colors**.

Although named colors provide 147 different options, this is a small amount when you consider the flexibility of CSS. To take advantage of the full spectrum of colors that CSS supports, you have the option of using *RGB colors*.

A screenshot of a code editor window with two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing a CSS rule for the 'h1' selector. The code is as follows:

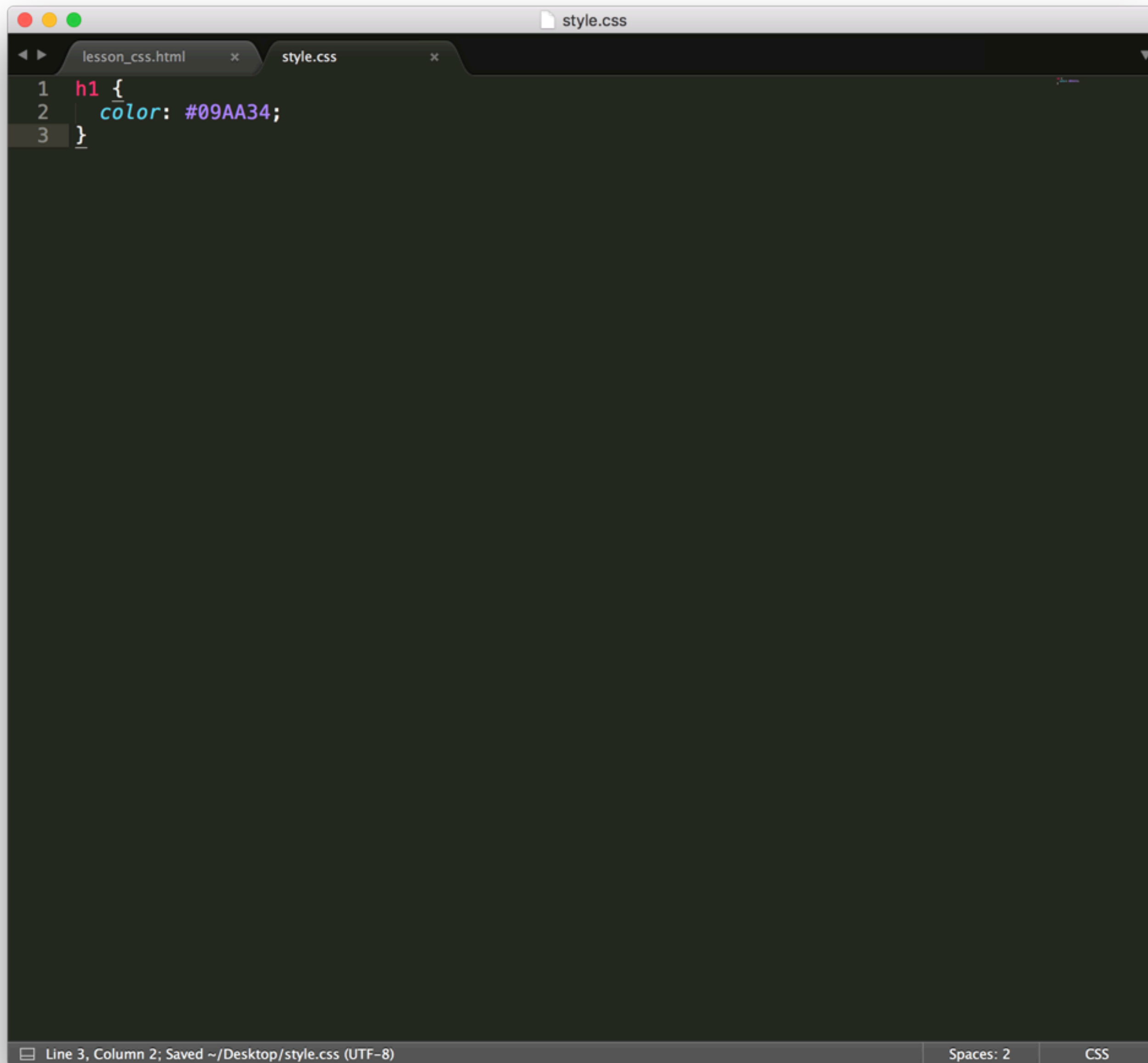
```
1 h1 {  
2   color: rgb(123, 20, 233);  
3   background-color: rgb(99, 21, 127);  
4 }
```

The code is syntax-highlighted: 'h1' is red, '{' is blue, 'color:' is blue, 'rgb(123, 20, 233);' is green, 'background-color:' is blue, 'rgb(99, 21, 127);' is green, and '}' is blue. The editor has a dark theme. At the bottom, a status bar shows 'Line 4, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

To use RGB colors, you can use the **rgb()** value when styling a color.

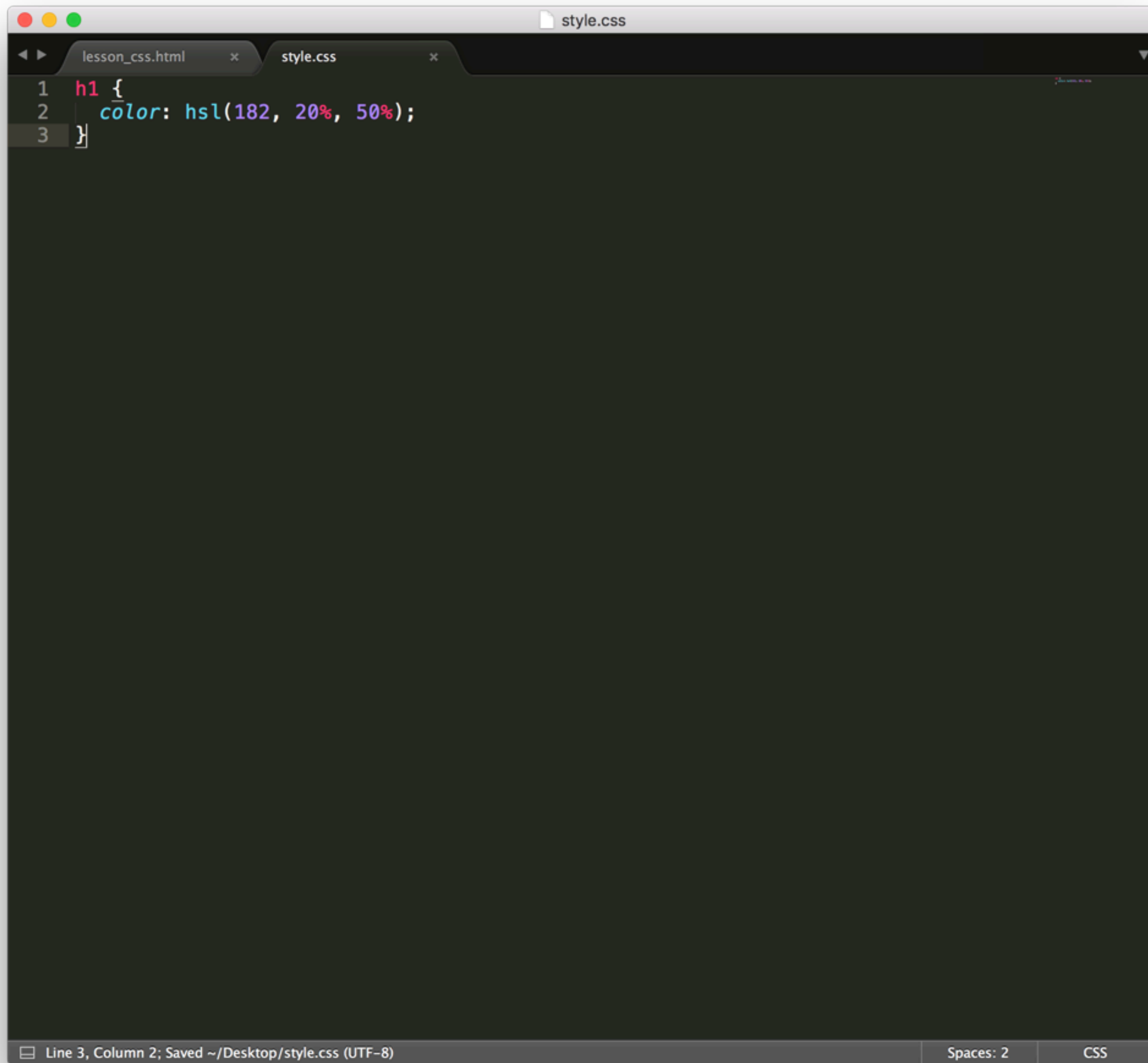
In this example, the value of **color** is set to **rgb()**. The three numbers in the parentheses represent the values for R, G, and B, in that order. Note that you can use **rgb()** for background colors as well.

There's an additional way to specify colors in
CSS: *hexadecimal color codes*, often referred to as
"hex color codes" for short.

A screenshot of a code editor window with two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code. Line 1: 'h1 {'; Line 2: 'color: #09AA34;'; Line 3: '}'. The code is syntax-highlighted. The status bar at the bottom indicates 'Line 3, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

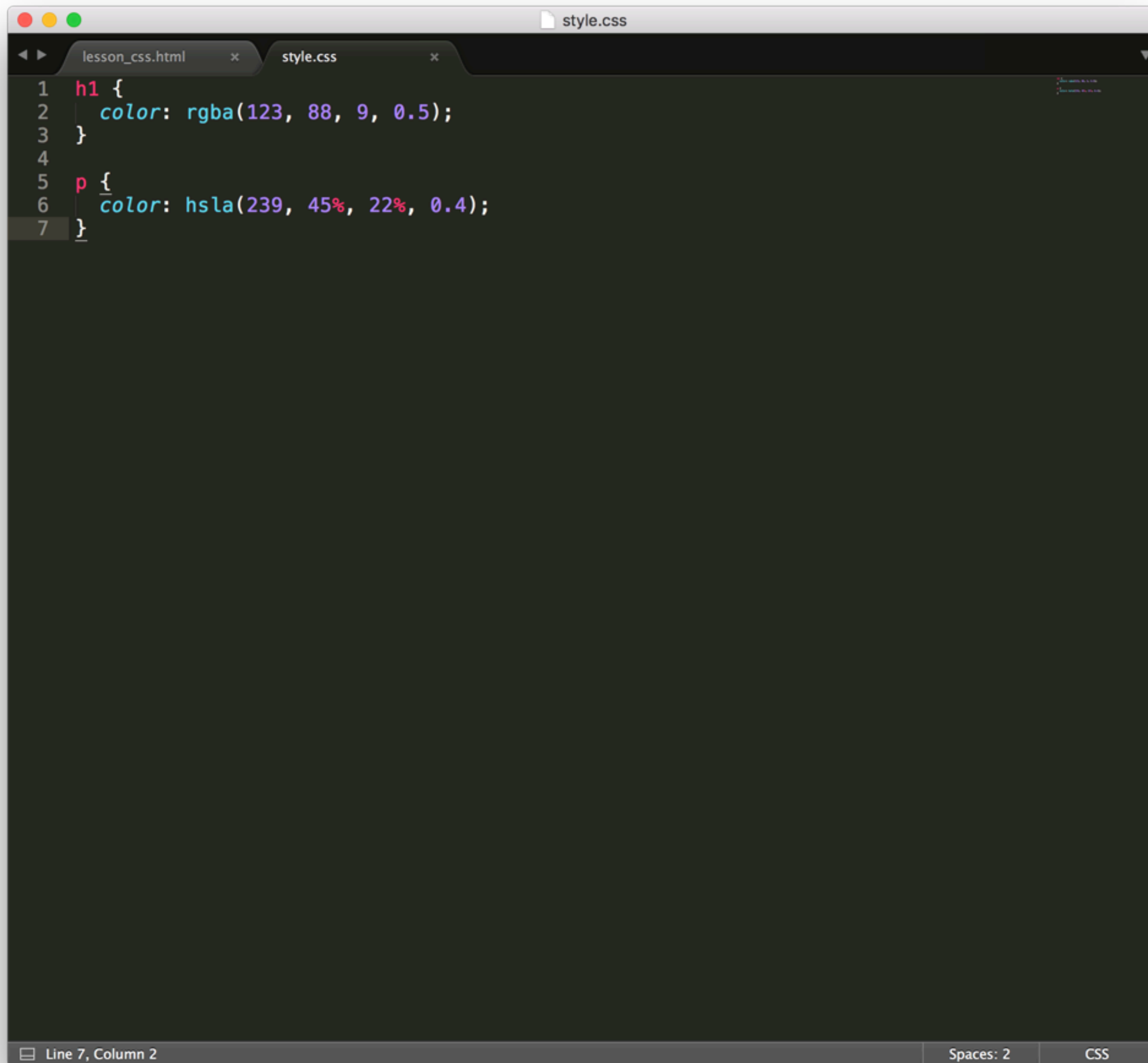
```
1 h1 {  
2   color: #09AA34;  
3 }
```

When read from left to right, each group of two characters responds to a value for red, green and blue, respectively. In the example above, **09** refers to the value for red, **AA** refers to the value for green, and **34** refers to the value for blue. All hex color codes begin with a **#** character.

A screenshot of a code editor window. The title bar shows 'style.css'. The editor has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code. Line 1: 'h1 {'; Line 2: 'color: hsl(182, 20%, 50%);'; Line 3: '}'. The code is syntax-highlighted. The bottom status bar shows 'Line 3, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

```
1 h1 {  
2   color: hsl(182, 20%, 50%);  
3 }
```

**Here's an example of hsl color.
Very similar to rgb syntax.**

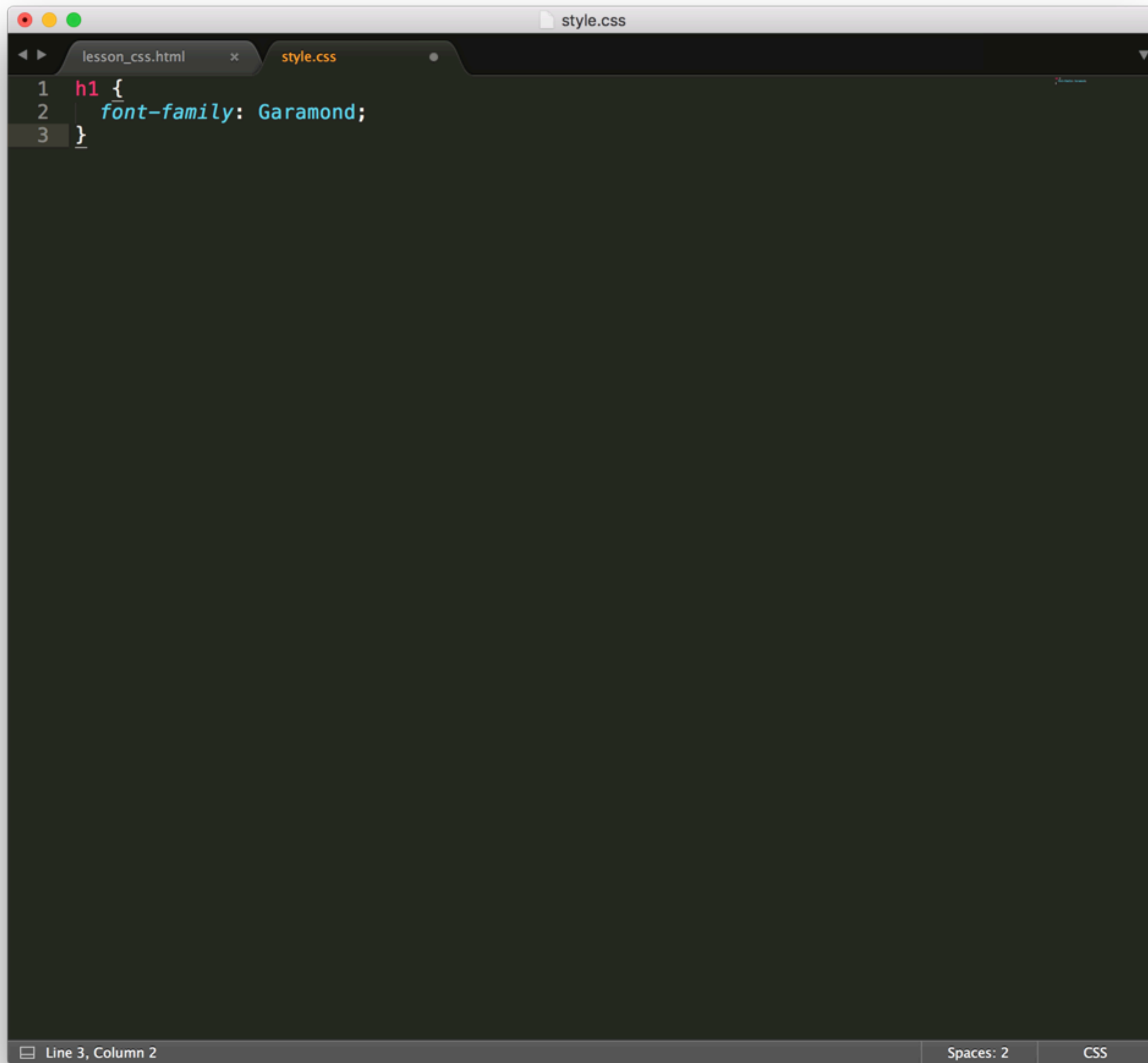
A screenshot of a code editor window with two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing CSS code for two selectors: 'h1' and 'p'. The 'h1' selector is on lines 1-3, and the 'p' selector is on lines 5-7. The 'p' selector's color property uses the 'hsla' function with values (239, 45%, 22%, 0.4). The editor has a dark theme and a status bar at the bottom showing 'Line 7, Column 2', 'Spaces: 2', and 'CSS'.

```
1 h1 {  
2   color: rgba(123, 88, 9, 0.5);  
3 }  
4  
5 p {  
6   color: hsla(239, 45%, 22%, 0.4);  
7 }
```

Opacity is a measure of how transparent a color is. To modify opacity in RGB colors, CSS offers the `rgba()` value. Note the slight difference in `rgb()` and `rgba()`.

The extra `a` character in the `rgba()` value is known as the *alpha value*. It represents the opacity of a color. The alpha value can be a number between 0 or 1, inclusive.

Now let's take a look at fonts and typography

A screenshot of a code editor window with a dark theme. The window has a title bar with three colored buttons (red, yellow, green) on the left and a tab labeled 'style.css' on the right. The editor shows three lines of CSS code: '1 h1 {', '2 font-family: Garamond;', and '3 }'. The text is color-coded: 'h1' is red, '{' is blue, 'font-family:' is green, 'Garamond;' is blue, and '}' is red. The status bar at the bottom shows 'Line 3, Column 2', 'Spaces: 2', and 'CSS'.

```
1 h1 {  
2 font-family: Garamond;  
3 }
```

To change the typeface of text on your web page, you can use the **font-family** property.

When setting typefaces on a web page, keep the following points in mind:

The font specified in a stylesheet must be installed on a user's computer in order for that font to display when a user visit the web page. We'll learn how to work around this issue soon.

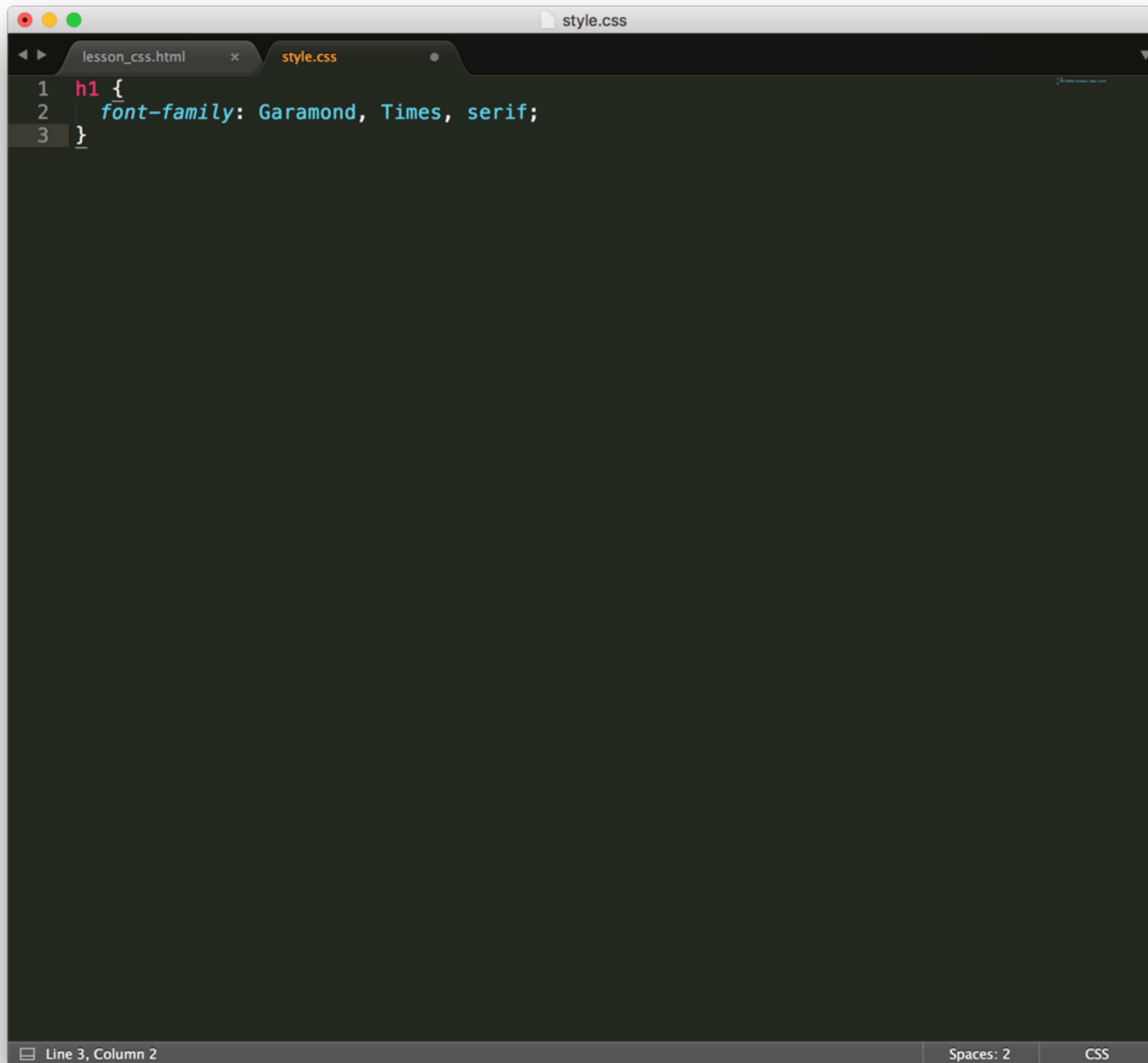
How exactly does the browser know what typeface to use when displaying the web page? The default typeface for all HTML elements is **Times New Roman**.

When the name of a typeface consists of more than one word, it must be enclosed in double quotes (otherwise it will not be recognized), like so:
"Courier New"

What happens when a font is not installed on a user's computer?

Most computers have a small set of typefaces pre-installed. This small set includes serif fonts and sans-serif fonts, like **Times New Roman** and **Arial**, respectively.

When the stylesheet specifies a font not installed on a user's computer, the pre-installed fonts can be used as *fallback fonts* for users.



A screenshot of a code editor window with a dark theme. The window has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing a CSS rule for the `h1` element. The rule is: `h1 { font-family: Garamond, Times, serif; }`. The code is syntax-highlighted: `h1` is red, `{` is blue, `font-family:` is green, `Garamond,` is blue, `Times,` is blue, `serif;` is green, and `}` is blue. The editor has a line number column on the left showing lines 1, 2, and 3. The status bar at the bottom shows 'Line 3, Column 2', 'Spaces: 2', and 'CSS'.

```
1 h1 {  
2   font-family: Garamond, Times, serif;  
3 }
```

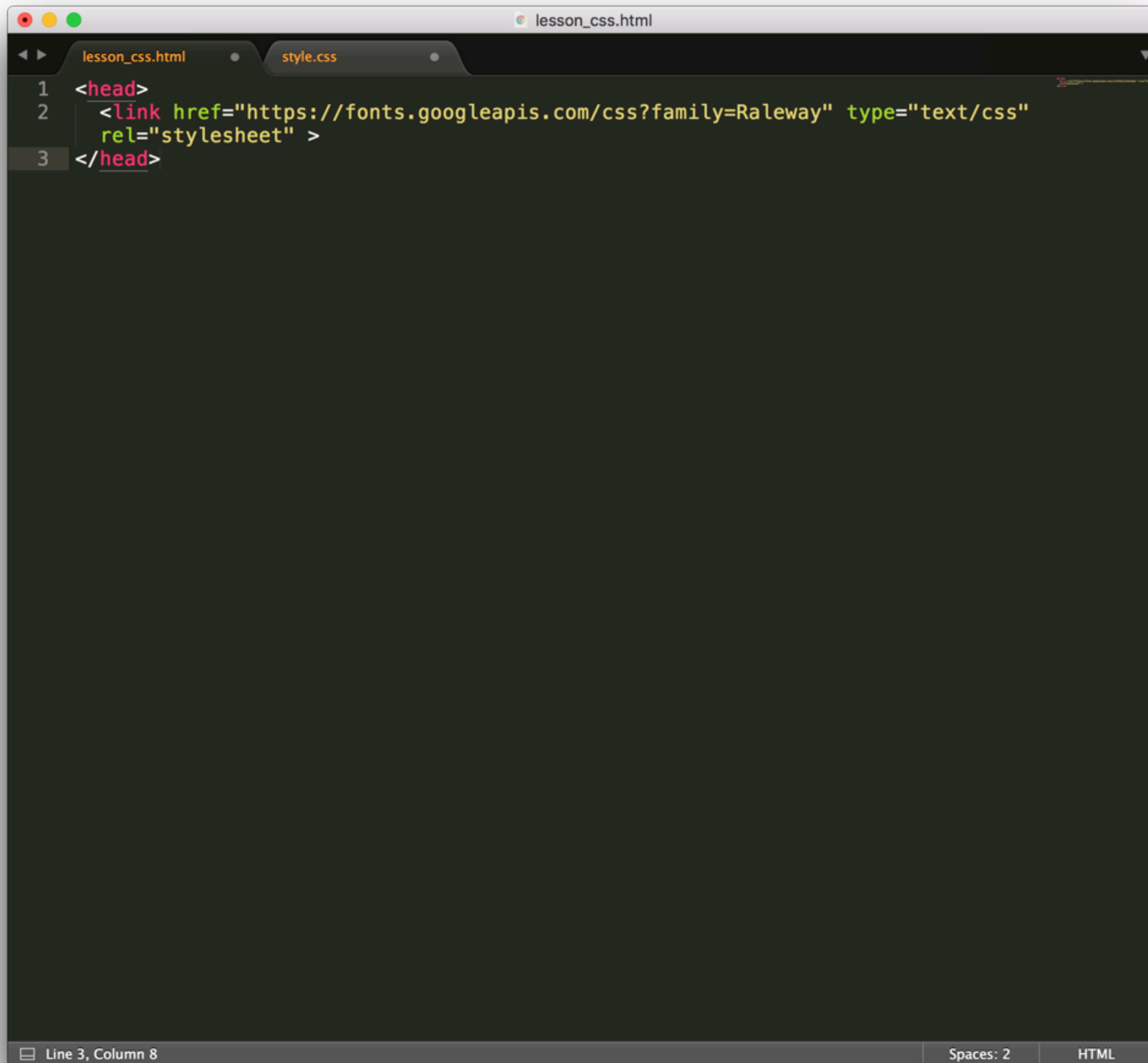
The CSS rule above says: "Use the **Garamond** font for all **<h1>** elements on the web page. If that font is not available, use the **Times** font. If both of those fonts are not available, use any **serif** font pre-installed on the user's computer." The fonts specified after **Garamond** are the fallback fonts.

It would be unrealistic to expect users to have all the fonts you might want to design with installed on their computers.

Fortunately, you don't have to predict which fonts are installed on a user's computer.

Google offers **Google Fonts**, a directory of thousands of open-source fonts that are free to use by anyone.

To use these fonts, you can link to a specific Google Font in your HTML code and use it immediately in your stylesheet. Because the HTML file links directly to the Google Font, a user's browser can display the typeface you specify. This avoids having to determine whether or not that font is installed on a user's computer.

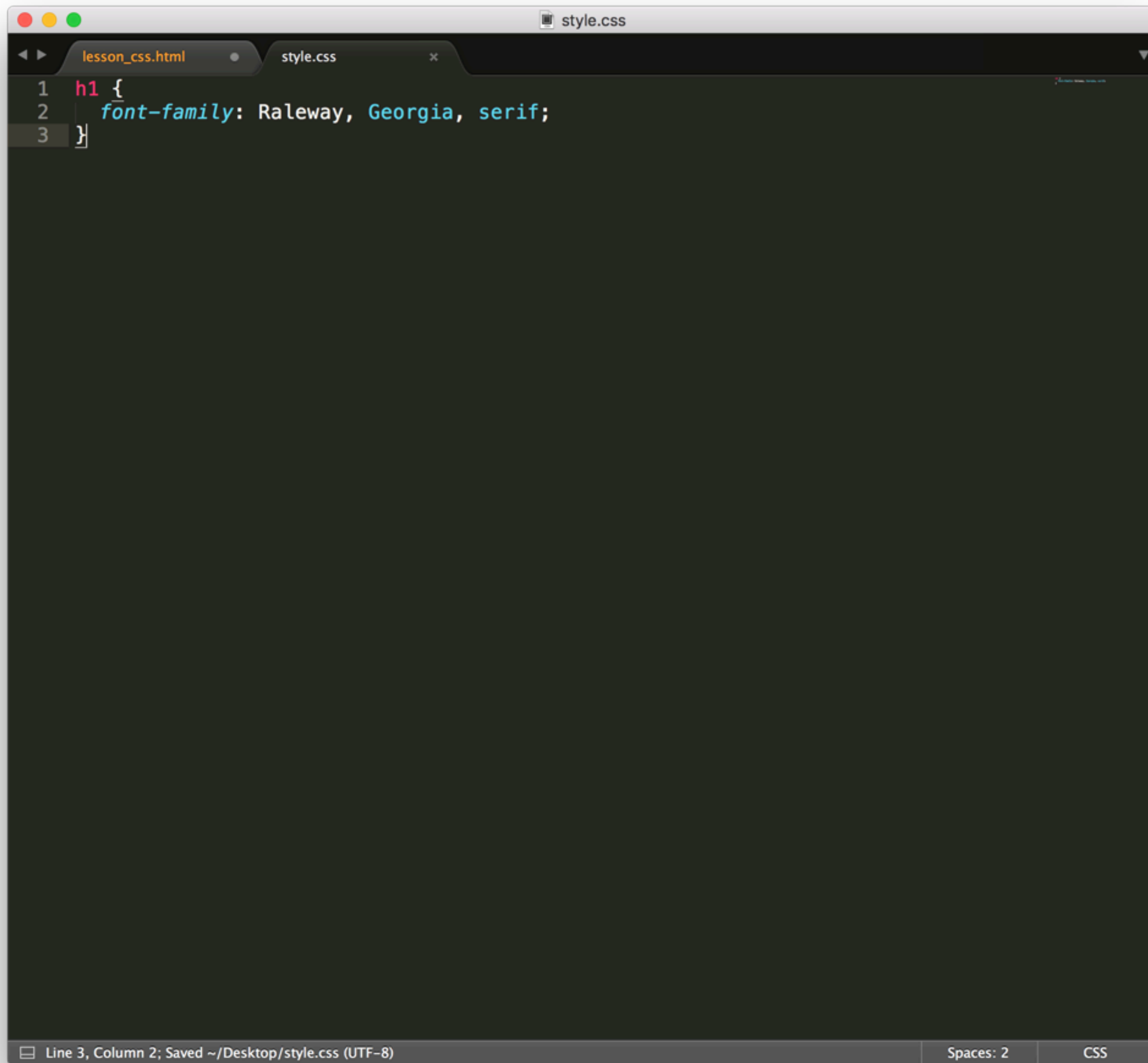
A screenshot of a code editor window titled 'lesson_css.html'. The editor shows three lines of HTML code in a dark theme. Line 1: <head>. Line 2: <link href="https://fonts.googleapis.com/css?family=Raleway" type="text/css" rel="stylesheet" >. Line 3: </head>. The 'href' attribute in line 2 is highlighted in green. The editor's status bar at the bottom shows 'Line 3, Column 8', 'Spaces: 2', and 'HTML'.

```
1 <head>
2   <link href="https://fonts.googleapis.com/css?family=Raleway" type="text/css"
3   rel="stylesheet" >
4 </head>
```

The **href** attribute is set to the following URL, which was retrieved from Google Fonts:

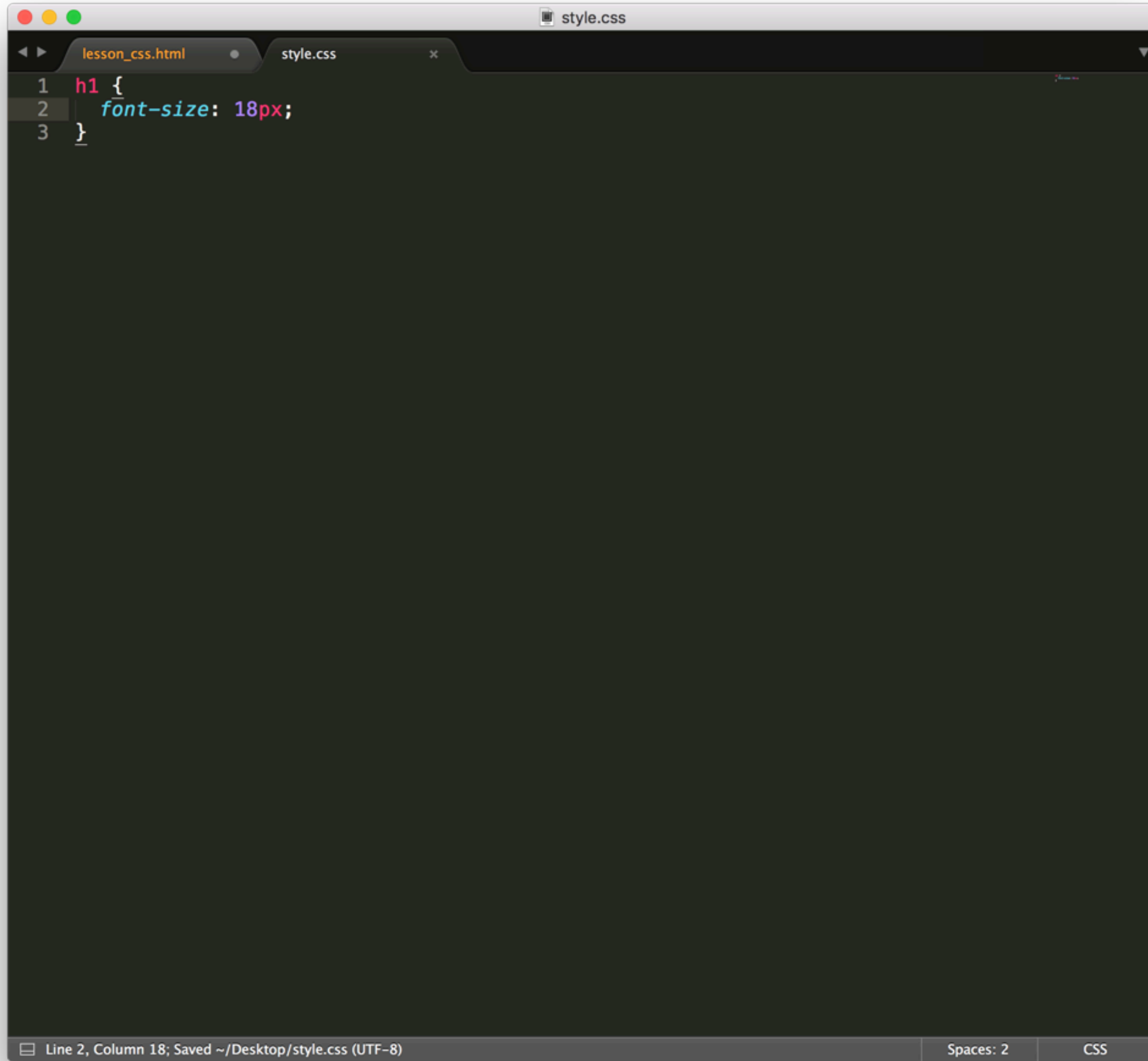
<https://fonts.googleapis.com/css?family=Raleway>

The URL in the example above specifies the **Raleway** typeface from Google Fonts.

A screenshot of a code editor window. The title bar shows 'style.css'. The editor has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code: '1 h1 {', '2 font-family: Raleway, Georgia, serif;', and '3 }'. The code is syntax-highlighted. The bottom status bar shows 'Line 3, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

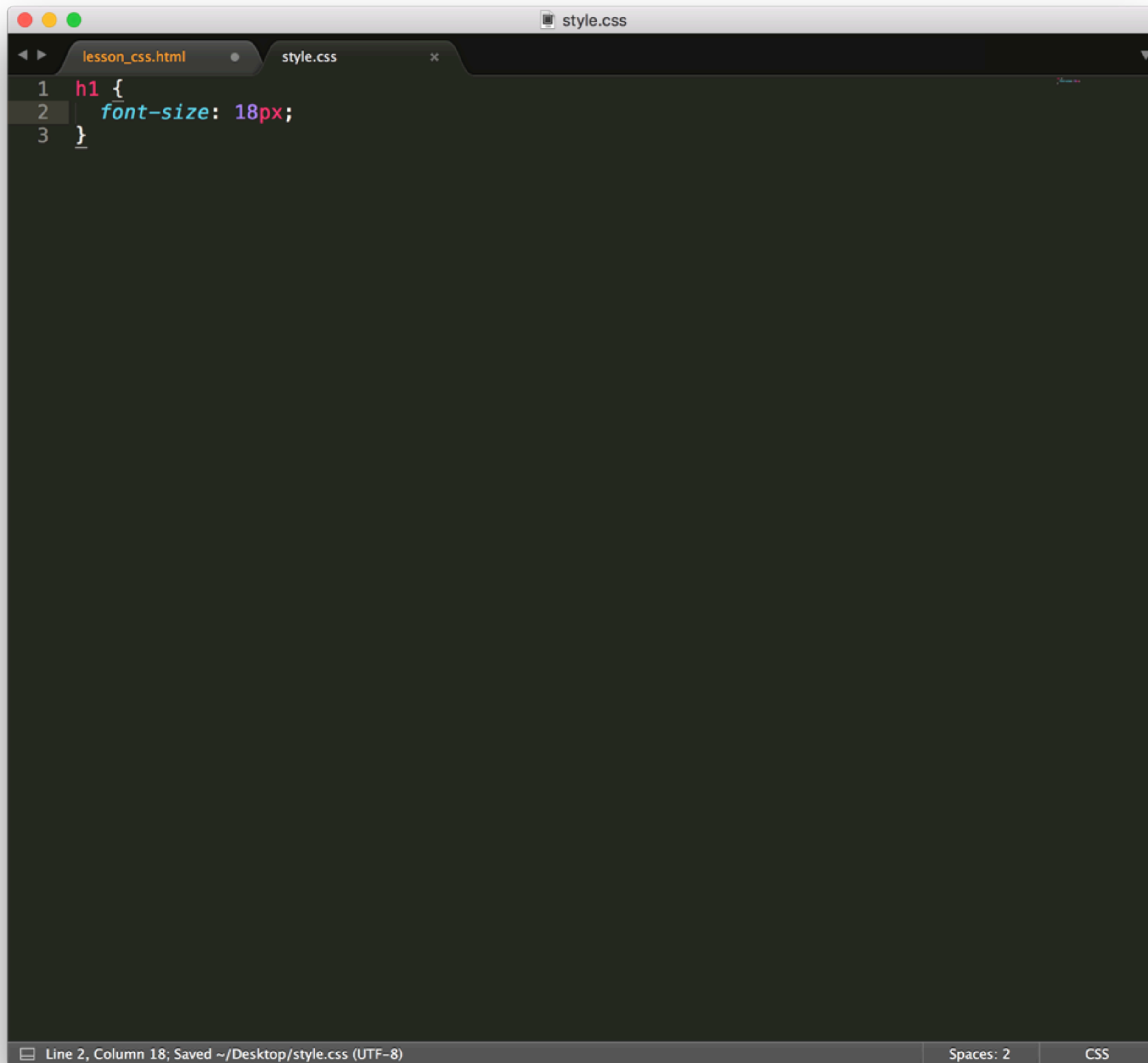
```
1 h1 {  
2 font-family: Raleway, Georgia, serif;  
3 }
```

You can now use this just as you would any other font.

A screenshot of a code editor window. The title bar shows 'style.css'. The editor has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code: '1 h1 {', '2 font-size: 18px;', and '3 }'. The code is syntax-highlighted. The status bar at the bottom shows 'Line 2, Column 18; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

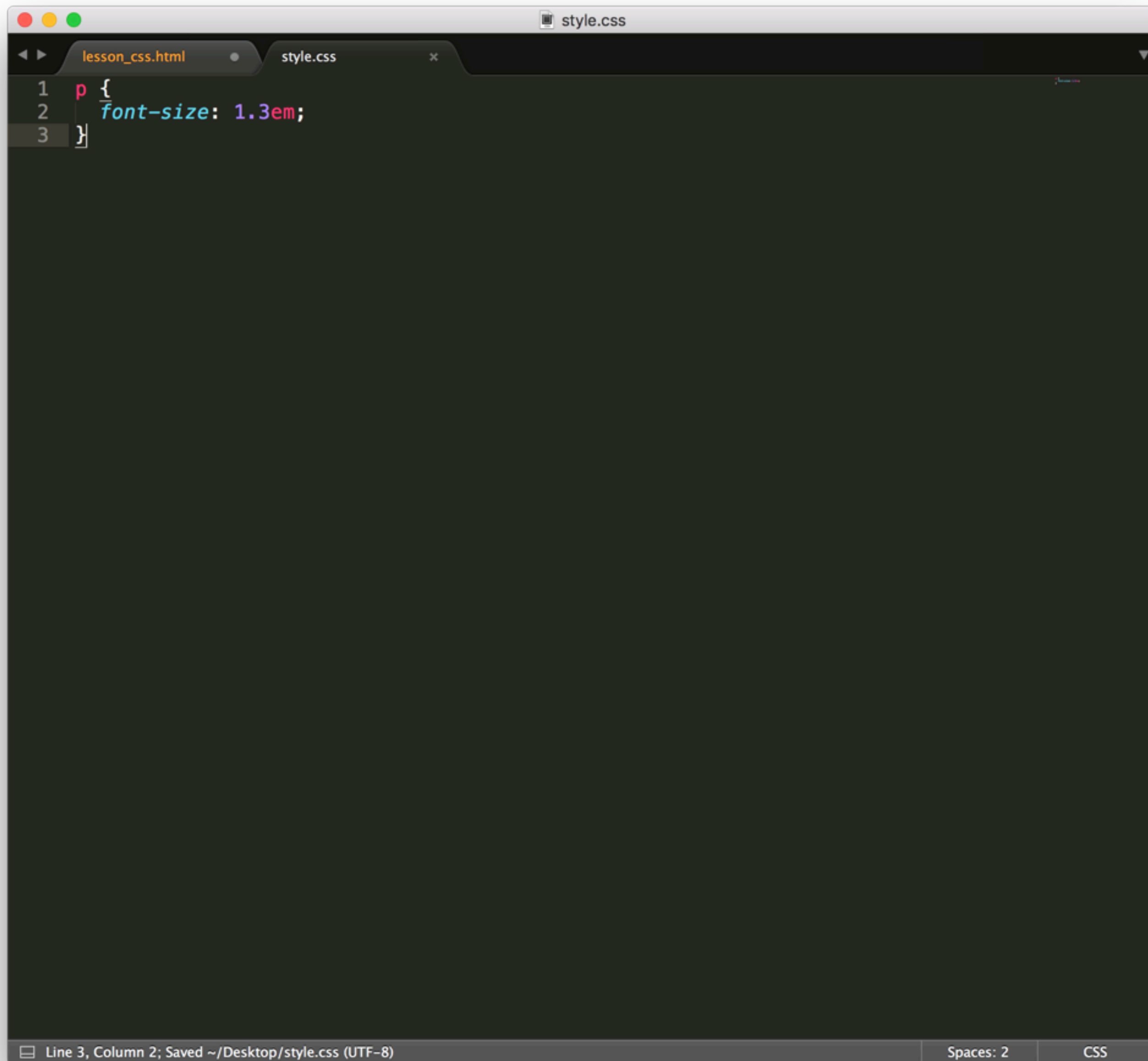
```
1 h1 {  
2 font-size: 18px;  
3 }
```

To change the size of text on your web page, you can use the **font-size**

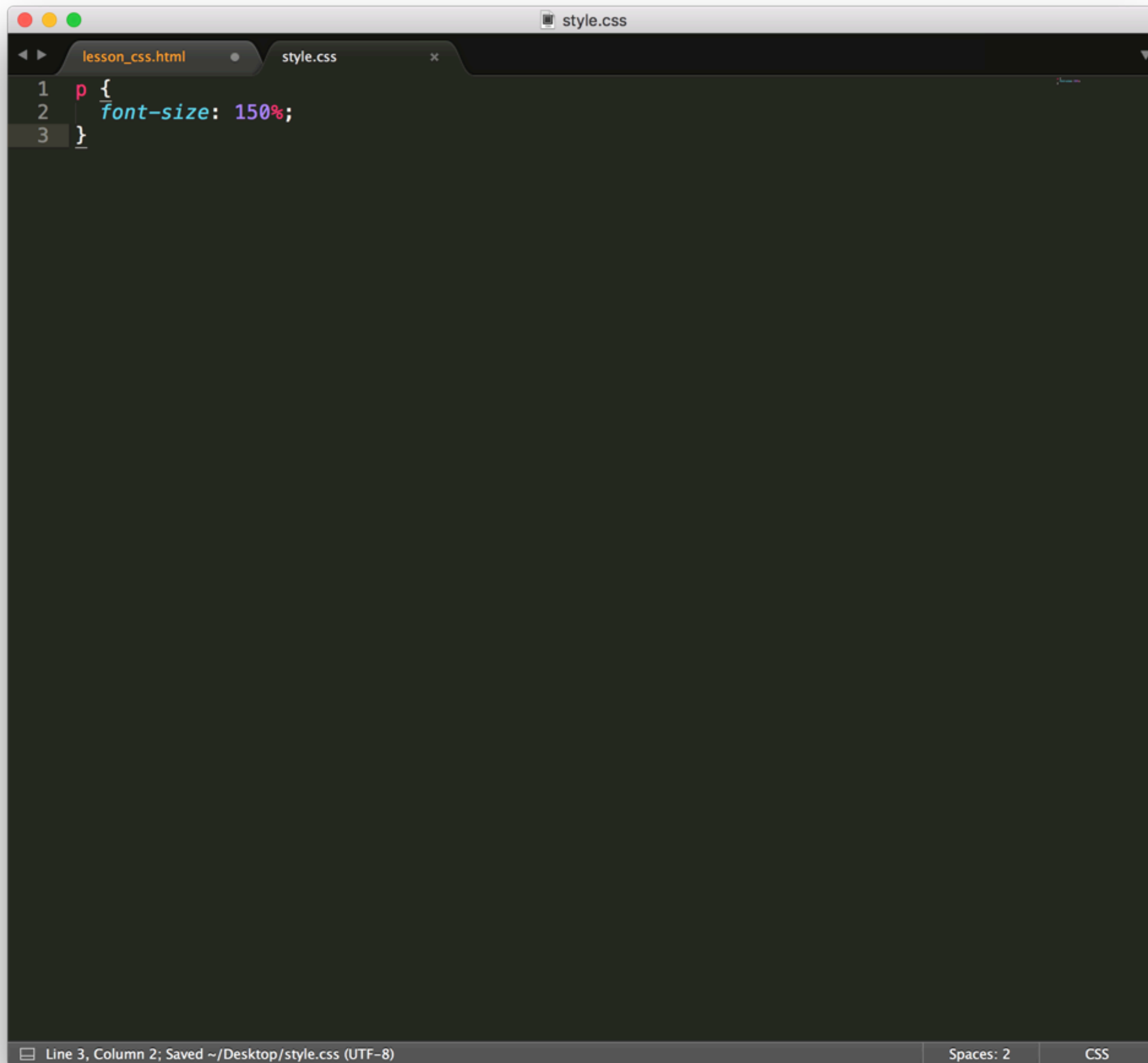
A screenshot of a code editor window. The title bar shows 'style.css'. The editor has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code: '1 h1 {', '2 font-size: 18px;', and '3 }'. The code is syntax-highlighted. The status bar at the bottom indicates 'Line 2, Column 18; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

There are three units of measurement for font size:

px – Represents the unit of *pixels*. The display of a computer monitor can be measured in pixels. A pixel is a small point on the display. Most computer monitors have a resolution of 72 pixels per inch, so a pixel represents about 1/72nd of an inch. Pixels are sometimes also referred to as *points*. Pixels are used to set the exact size of an element.

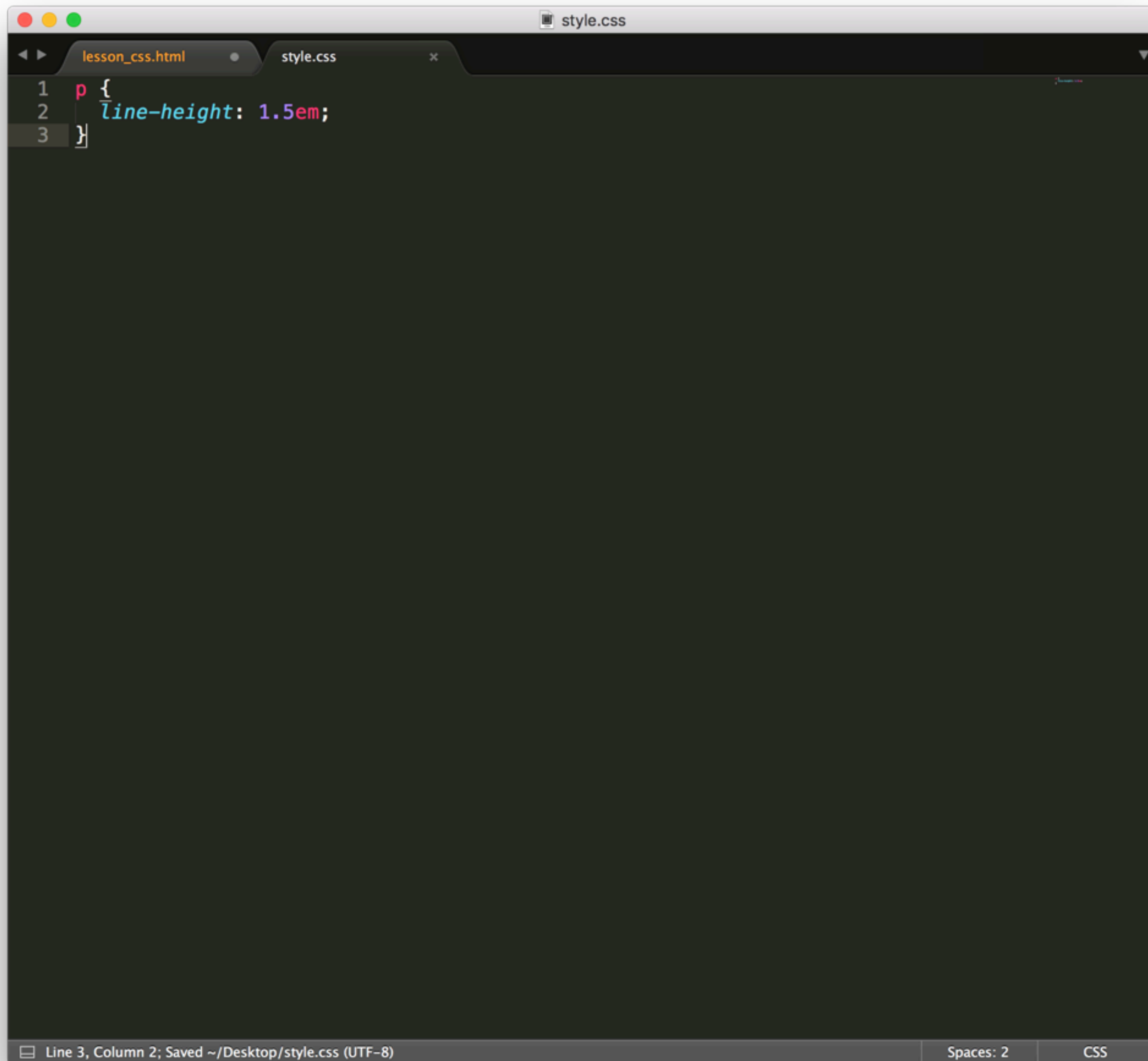
A screenshot of a code editor window. The title bar shows 'style.css'. The editor has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code: '1 p {', '2 font-size: 1.3em;', and '3 }'. The code is syntax-highlighted. The status bar at the bottom indicates 'Line 3, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

em – Pronounced just as it looks, "em." An em is equal to the width of the letter "m". Ems are a relative unit of measurement. They change the size of text relative to the parent element's size of text.

A screenshot of a code editor window with a dark theme. The window has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code. Line 1: 'p {'; Line 2: 'font-size: 150%;'; Line 3: '}'. The status bar at the bottom indicates 'Line 3, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

```
1 p {  
2   font-size: 150%;  
3 }
```

% – Percentages are also a relative unit of measurement. The default size of text in web browsers is 16 pixels, or **16px**. When percentages are used, they set the size of text relative to this default size. For example, setting the font size to **200%** would be equivalent to setting it to **32px**.

A screenshot of a code editor window. The title bar shows 'style.css'. The editor has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing a CSS rule for the 'p' element. The code is:

```
1 p {  
2   line-height: 1.5em;  
3 }
```

 The status bar at the bottom indicates 'Line 3, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

You can modify the spacing between lines of text with the **line-height** property.

When the line height of an element is modified, you are manipulating the *leading* (pronounced "ledding") of the font. When the line height is increased, the spacing between lines of text increases.

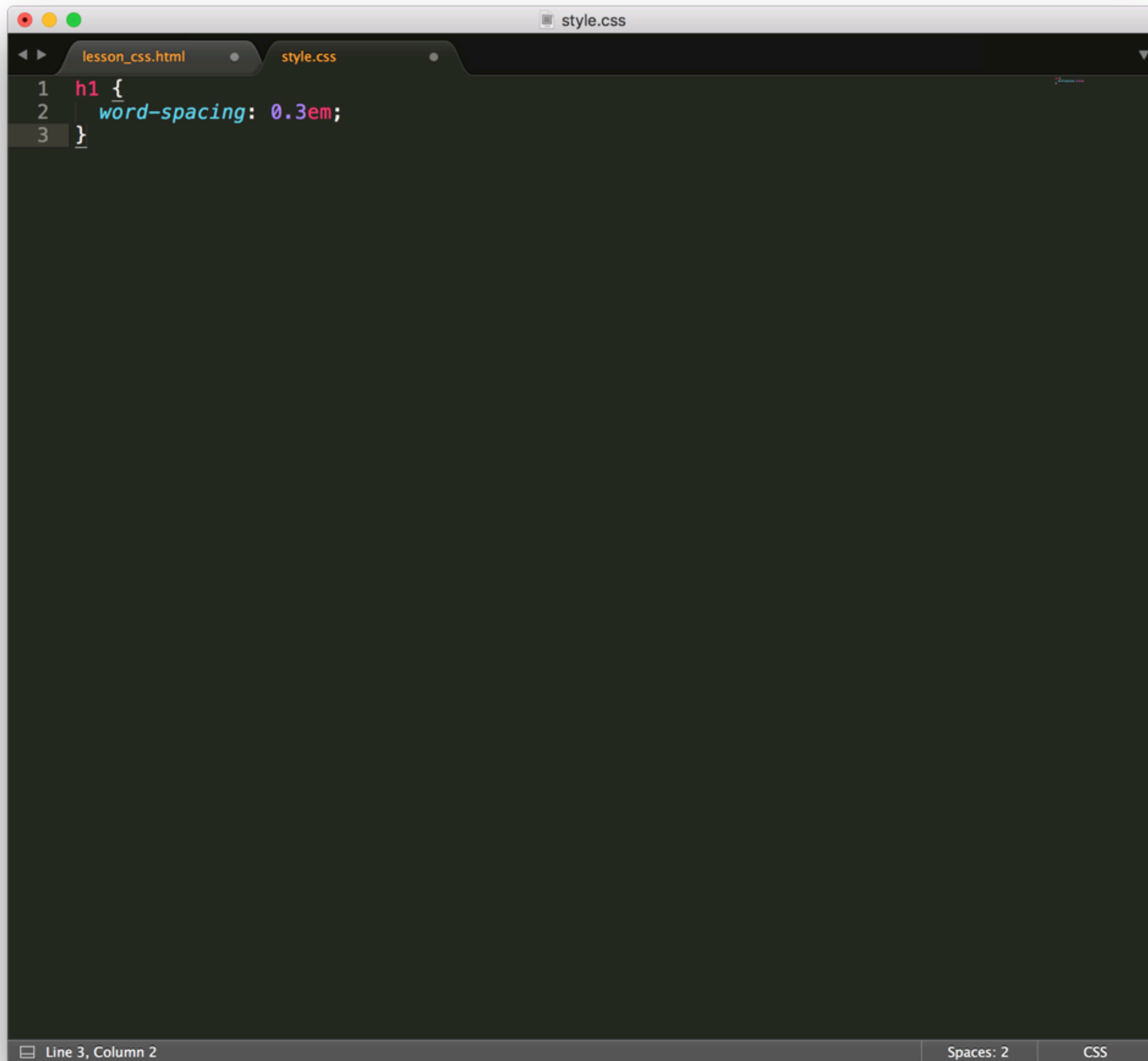
The line height can be modified using pixels or ems, but the unit of ems is preferred, since ems offer a spacing relative to the size of the text on the page.

The fastest cat

can race at 75

leading
font size

line height

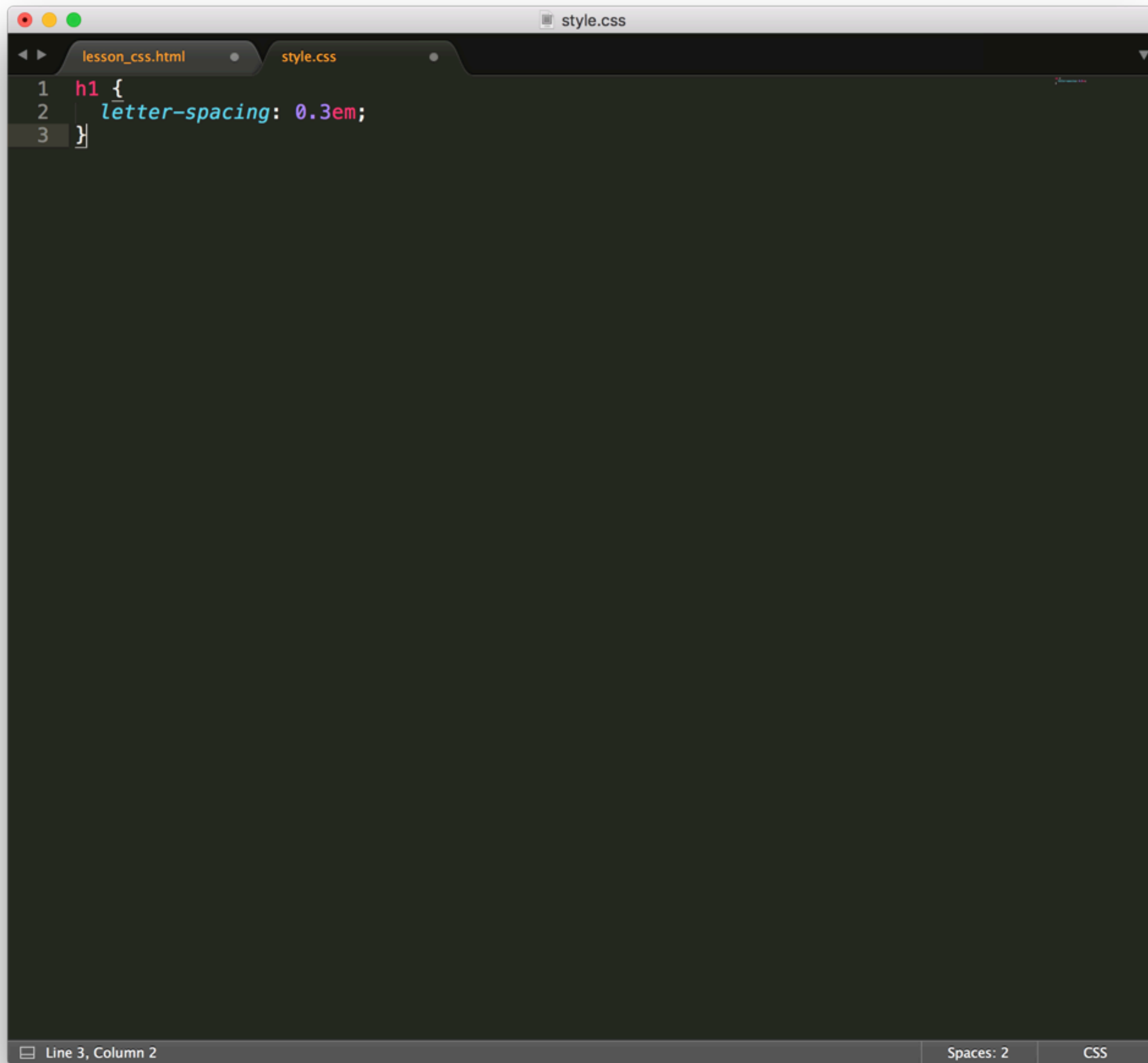
A screenshot of a code editor window with a dark theme. The window has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code. Line 1: 'h1 {'; Line 2: 'word-spacing: 0.3em;'; Line 3: '}'. The status bar at the bottom shows 'Line 3, Column 2', 'Spaces: 2', and 'CSS'.

```
1 h1 {  
2   word-spacing: 0.3em;  
3 }
```

You can also increase the spacing between words in a body of text, technically known as *word spacing*.

To do so, you can use the **word-spacing** property.

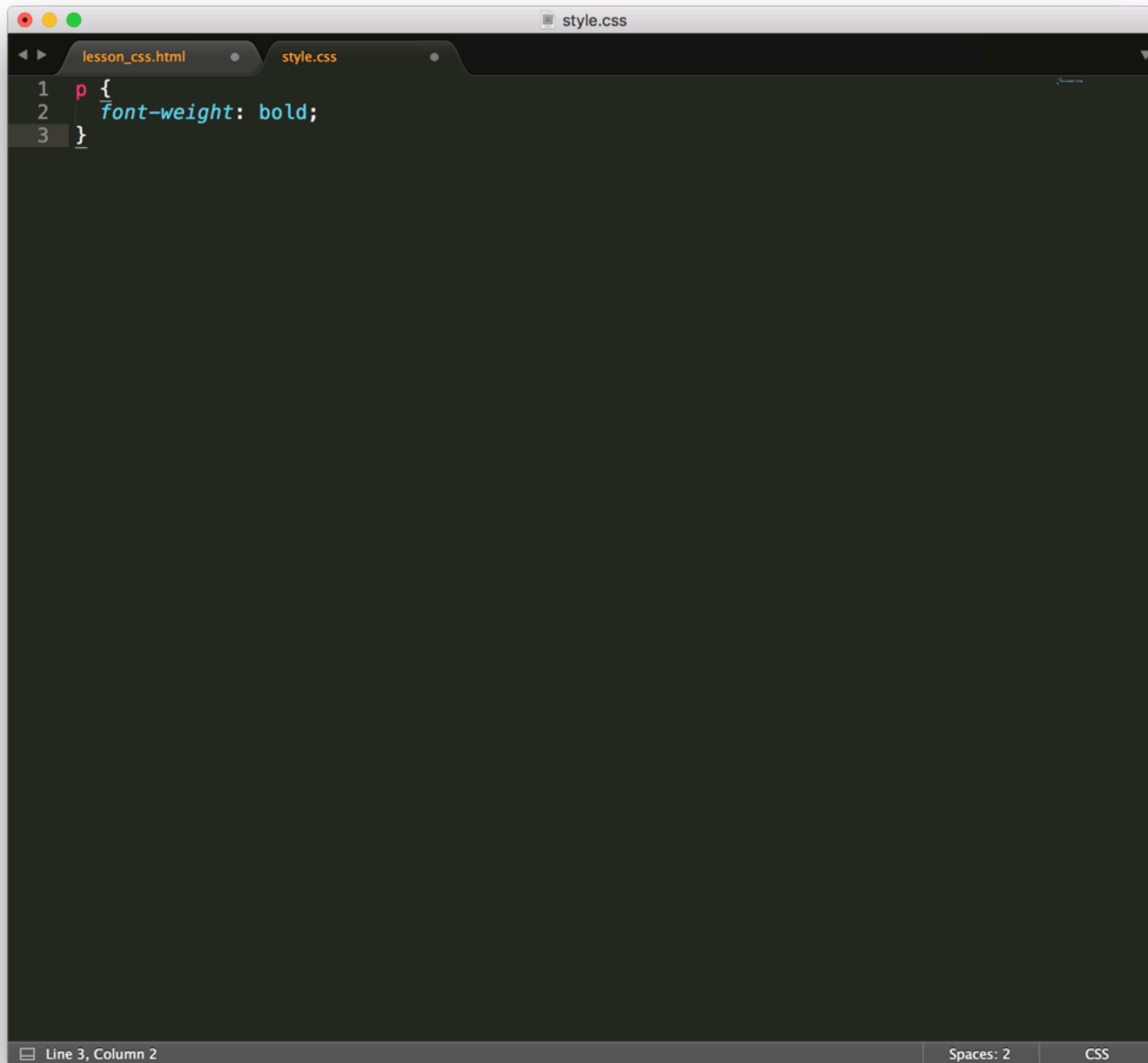
The default amount of space between words is usually **0.25em**. In this example above, the word spacing is set to **0.3em**, which represents an increase of only **.05em** in word spacing.



A screenshot of a code editor window with a dark theme. The window has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code. Line 1: `h1 {`, Line 2: `letter-spacing: 0.3em;`, Line 3: `}`. The code is color-coded: 'h1' is red, '{' is blue, 'letter-spacing' is blue, '0.3em' is green, and ';' is red. The editor has a status bar at the bottom showing 'Line 3, Column 2', 'Spaces: 2', and 'CSS'.

```
1 h1 {  
2 letter-spacing: 0.3em;  
3 }
```

The technical term for adjusting the spacing between letters is called "kerning". Kerning can be adjusted with the **letter-spacing** property in CSS

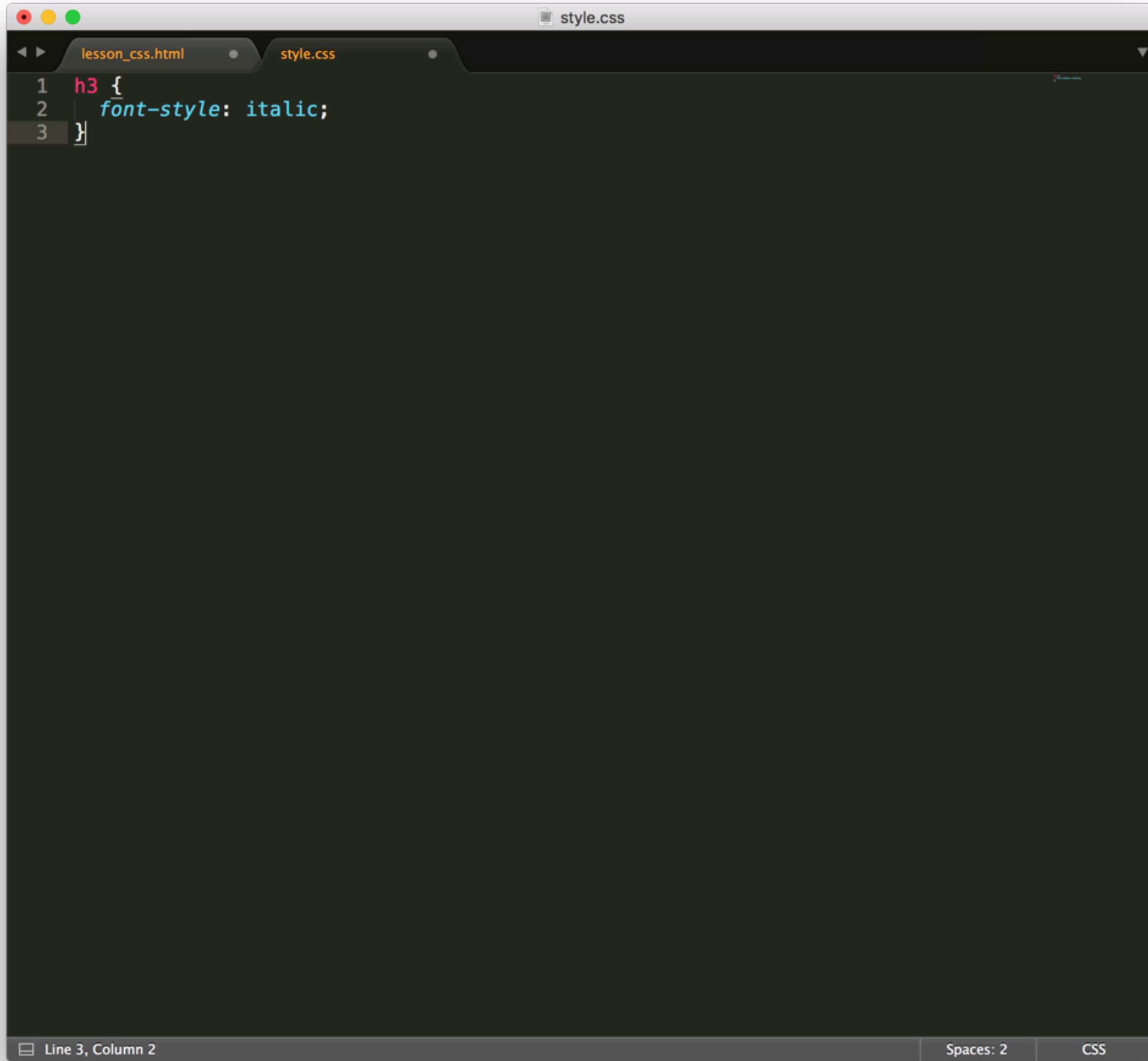
A screenshot of a code editor window with two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code. Line 1: 'p {' (with a red 'p' tag indicator). Line 2: 'font-weight: bold;' (with a blue 'font-weight' property indicator). Line 3: '}' (with a blue closing brace indicator). The editor has a dark background and a light-colored border. The status bar at the bottom shows 'Line 3, Column 2', 'Spaces: 2', and 'CSS'.

```
1 p {  
2   font-weight: bold;  
3 }
```

font-weight property turns bold on or off.

The **font-weight** property has a second value: **normal**. Why does it exist?

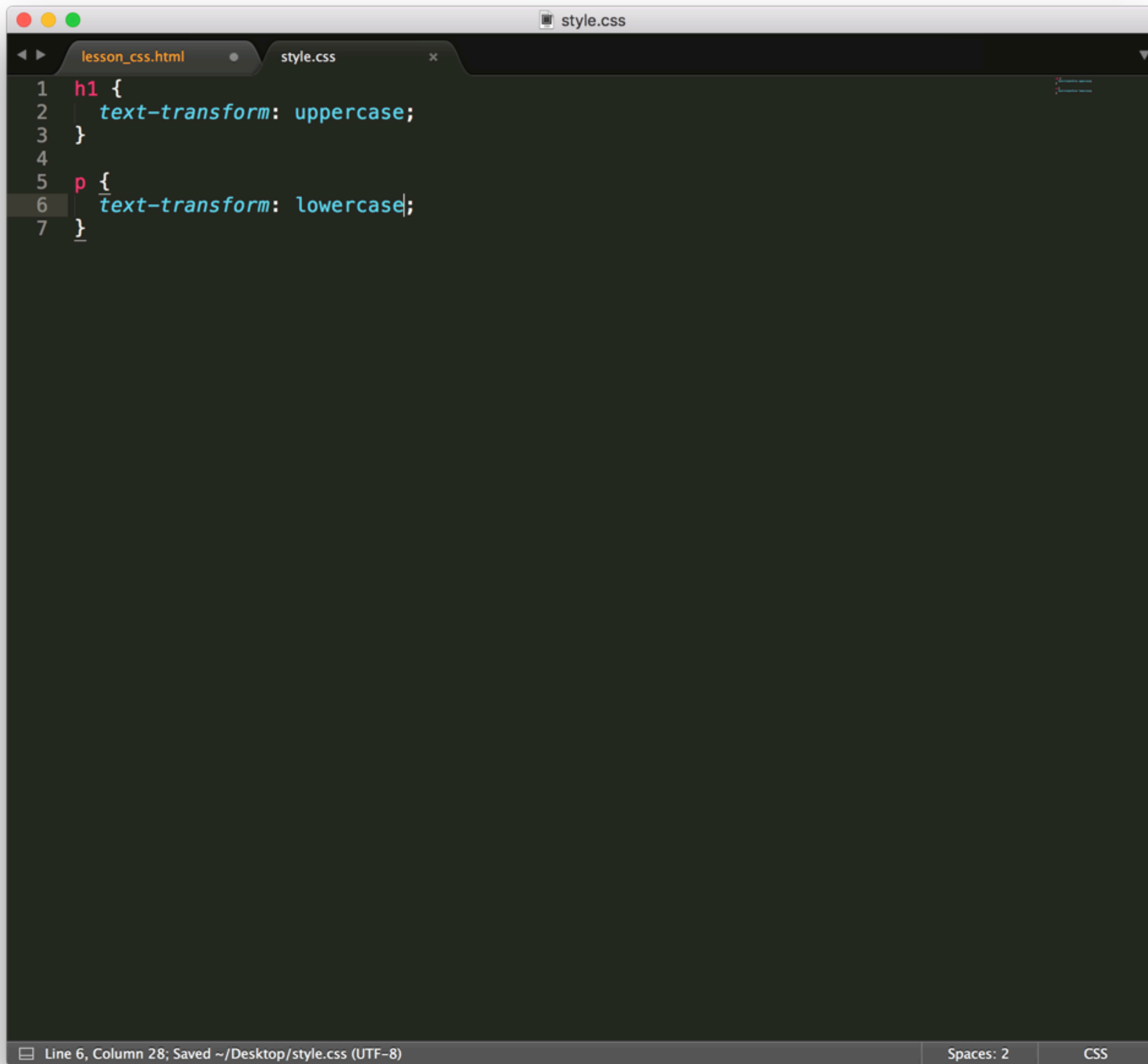
If we wanted *all* text on a web page to appear bolded, we could select all text elements and change their font weight to **bold**. If a certain section of text was required to appear normal, however, we could set the font weight of that particular element to **normal**, essentially "shutting off" bold for that element.



A screenshot of a code editor window with a dark theme. The window has a title bar with three colored buttons (red, yellow, green) and a tab labeled 'style.css'. The editor shows three lines of CSS code: line 1 is 'h3 {', line 2 is 'font-style: italic;', and line 3 is '}'. The code is syntax-highlighted: 'h3' is red, '{' is blue, 'font-style' is light blue, 'italic;' is green, and '}' is blue. The status bar at the bottom shows 'Line 3, Column 2', 'Spaces: 2', and 'CSS'.

```
1 h3 {  
2   font-style: italic;  
3 }
```

You can also *italicize* words with the **font-style** property

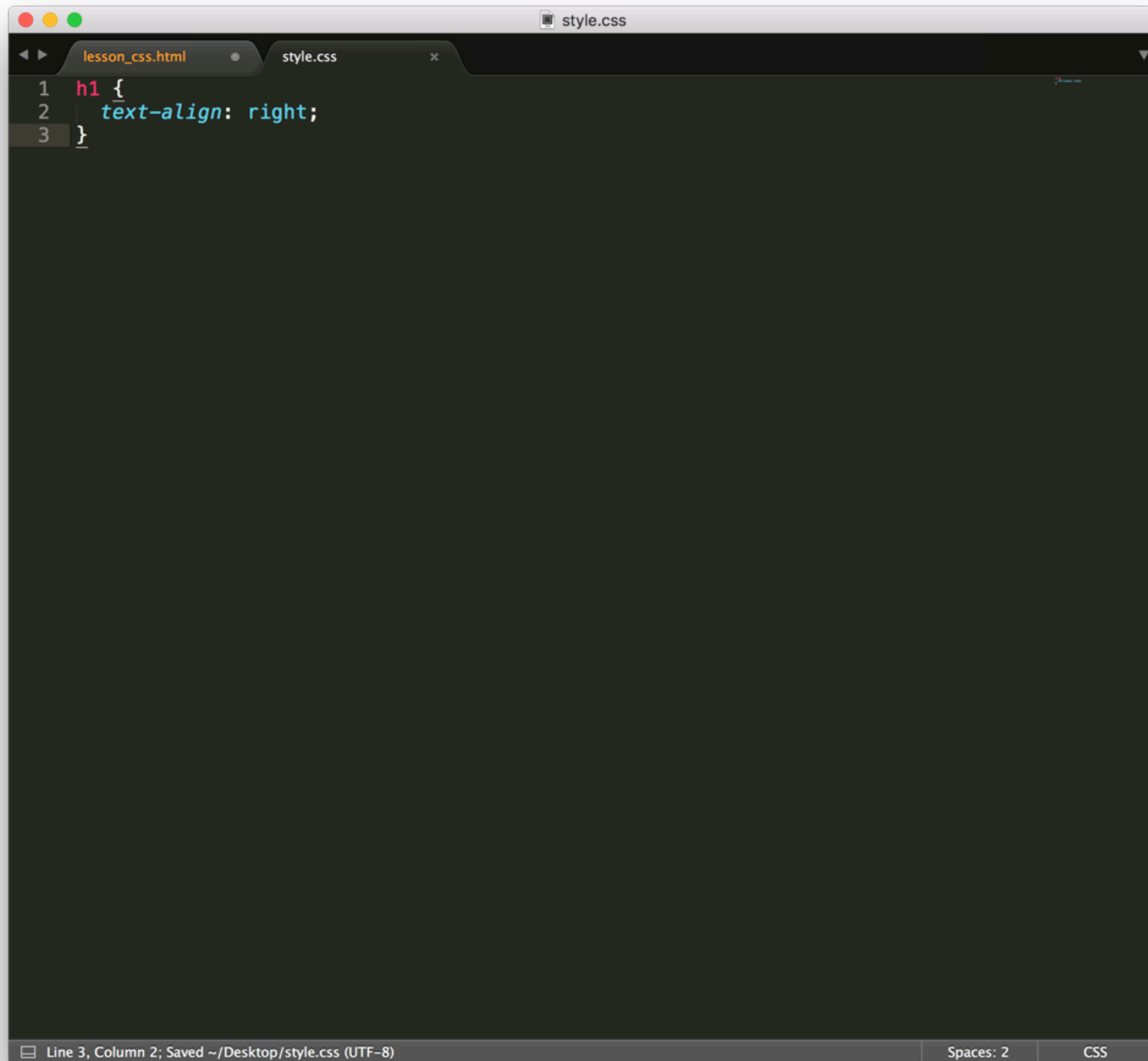


A screenshot of a code editor window with a dark theme. The window has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing the following CSS code:

```
1 h1 {  
2   text-transform: uppercase;  
3 }  
4  
5 p {  
6   text-transform: lowercase;  
7 }
```

The status bar at the bottom indicates 'Line 6, Column 28; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

Text can also be styled to appear in either all uppercase or lowercase with the **text-transform** property.

A screenshot of a code editor window with a dark theme. The window has two tabs: 'lesson_css.html' and 'style.css'. The 'style.css' tab is active, showing three lines of CSS code. Line 1: 'h1 {'; Line 2: 'text-align: right;'; Line 3: '}'. The code is syntax-highlighted. The status bar at the bottom shows 'Line 3, Column 2; Saved ~/Desktop/style.css (UTF-8)', 'Spaces: 2', and 'CSS'.

```
1 h1 {  
2   text-align: right;  
3 }
```

To move, or align, text, we can use the **text-align** property.

The **text-align** property can be set to one of the following three values:

left – aligns text to the left hand side of the browser.

center – centers text.

right – aligns text to the right hand side of the browser.