

jQuery

**jQuery is a library, or set of helpful add-ons, to the
JavaScript programming language**


jQuery is much better at giving you immediate, visual results than regular JavaScript.

You'll most likely be using a mix of basic JavaScript and jQuery together to get the effects you want.


jQuery

Secure https://jquery.com

Eric




write less, do more.



Your donations help fund the continued development and growth of jQuery.


[SUPPORT THE PROJECT](#)

[Download](#)[API Documentation](#)[Blog](#)[Plugins](#)[Browser Support](#)




Lightweight Footprint

Only 32kB minified and gzipped. Can also be included as an AMD module




CSS3 Compliant

Supports CSS3 selectors to find elements as well as in style property manipulation



Cross-Browser

[Chrome, Edge, Firefox, IE, Safari, Android, iOS, and more](#)



Download jQuery

v3.2.1





The 1.x and 2.x branches no longer receive patches.

[View Source on GitHub](#) →
[How jQuery Works](#) →

What is jQuery?


jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

Corporate Members



Resources

- [jQuery Core API Documentation](#)
- [jQuery Learning Center](#)
- [jQuery Blog](#)
- [Contribute to jQuery](#)
- [About the jQuery Foundation](#)
- [Browse or Submit jQuery Bugs](#)

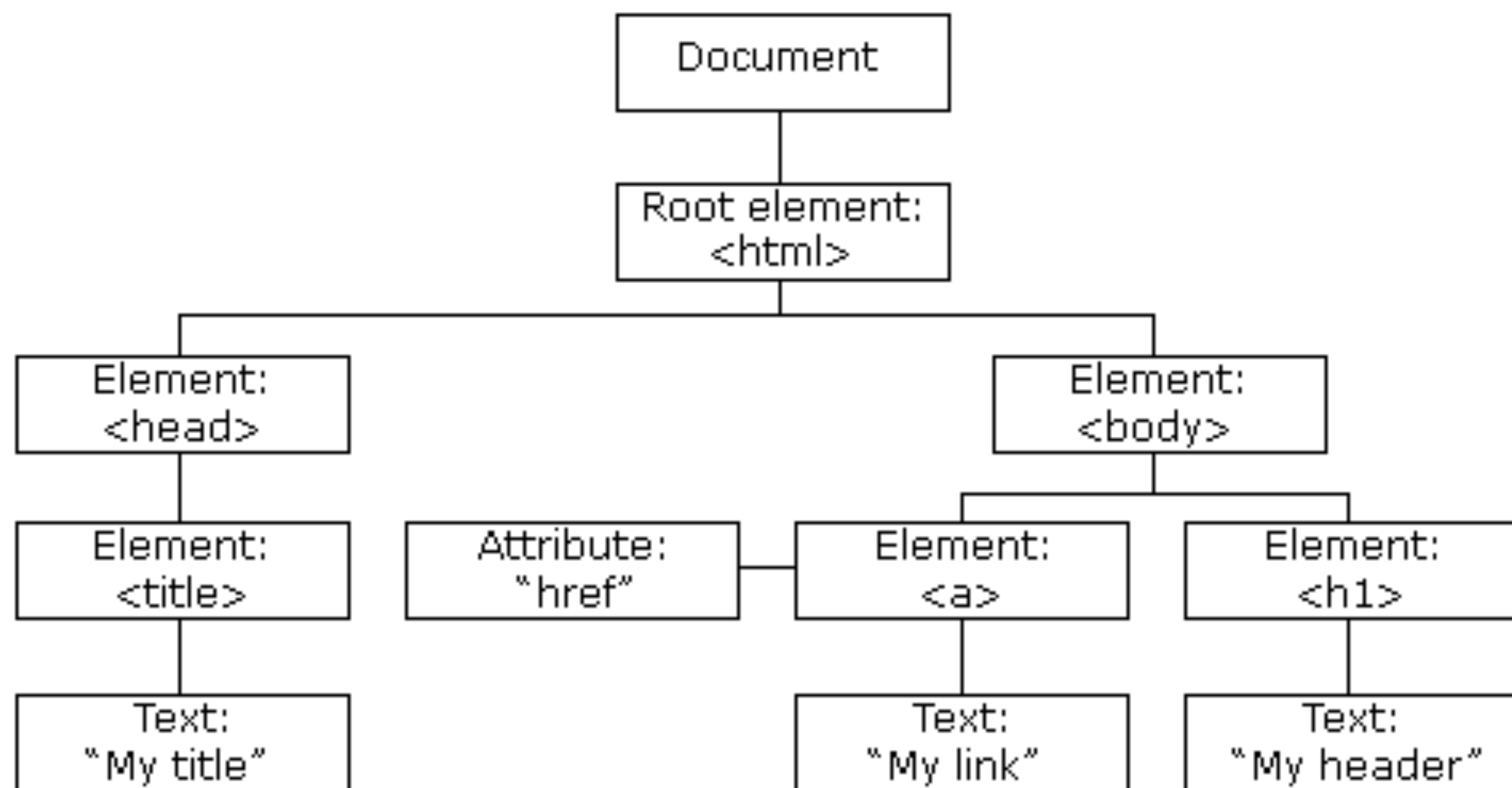


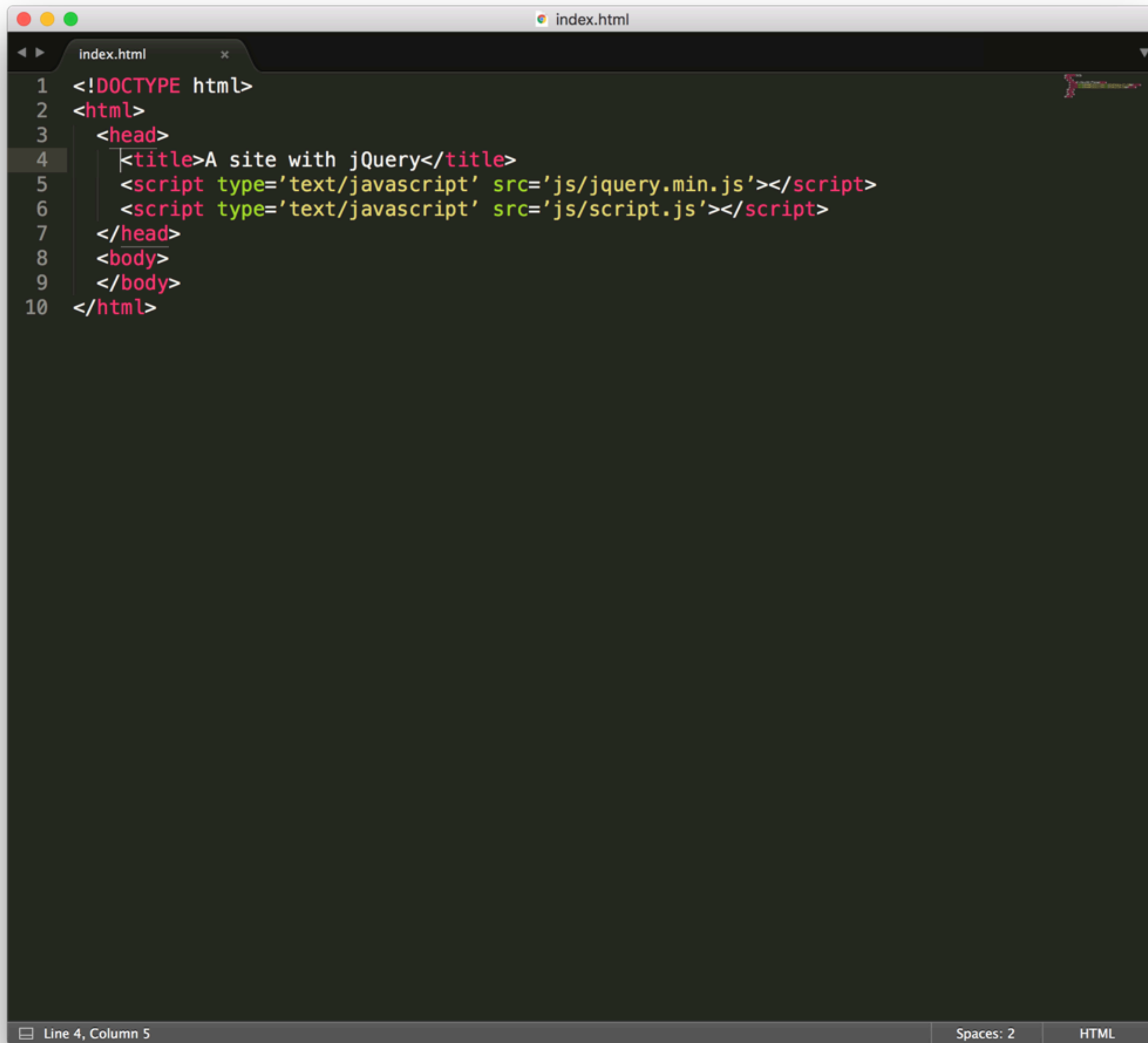
<https://github.com/jquery/jquery/issues>

**To get the most out of jQuery, we should review
how an HTML page is put together.**

An HTML document is structured according to the Document Object Model, or DOM. It's by interacting with the DOM that jQuery is able to access and modify HTML.

The DOM consists of every element on the page, laid out in a hierarchical way that reflects the way the HTML document is ordered.





```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>A site with jQuery</title>
5     <script type='text/javascript' src='js/jquery.min.js'></script>
6     <script type='text/javascript' src='js/script.js'></script>
7   </head>
8   <body>
9   </body>
10 </html>
```

The image shows a code editor window titled 'index.html'. The code is an HTML document structure. Line 4 is highlighted. The status bar at the bottom indicates 'Line 4, Column 5', 'Spaces: 2', and 'HTML'.

You can download and add
jQuery in the head of the html
document.

Hosted Libraries | Hosted Lib x

Eric

Secure

https://developers.google.com/speed/libraries/

☆

I

off

Google Hosted Libraries

Search

ALL PRODUCTS

jQuery

3.x snippet:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js">
</script>
```

2.x snippet:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.4/jquery.min.js">
</script>
```

1.x snippet:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js">
</script>
```

site:

jquery.com

versions:

3.2.1, 3.2.0, 3.1.1, 3.1.0, 3.0.0, 2.2.4, 2.2.3, 2.2.2, 2.2.1, 2.2.0, 2.1.4, 2.1.3, 2.1.1, 2.1.0, 2.0.3, 2.0.2, 2.0.1, 2.0.0, 1.12.4, 1.12.3, 1.12.2, 1.12.1, 1.12.0, 1.11.3, 1.11.2, 1.11.1, 1.11.0, 1.10.2, 1.10.1, 1.10.0, 1.9.1, 1.9.0, 1.8.3, 1.8.2, 1.8.1, 1.8.0, 1.7.2, 1.7.1, 1.7.0, 1.6.4, 1.6.3, 1.6.2, 1.6.1, 1.6.0, 1.5.2, 1.5.1, 1.5.0, 1.4.4, 1.4.3, 1.4.2, 1.4.1, 1.4.0, 1.3.2, 1.3.1, 1.3.0, 1.2.6, 1.2.3

note:

Contents

Libraries

AngularJS

Angular Material

Angular UI Router

Dojo

Ext Core

Hammer.JS

Indefinite Observable

jQuery

jQuery Mobile

jQuery UI

MooTools

Prototype

script.aculo.us

Shaka Player

SPF

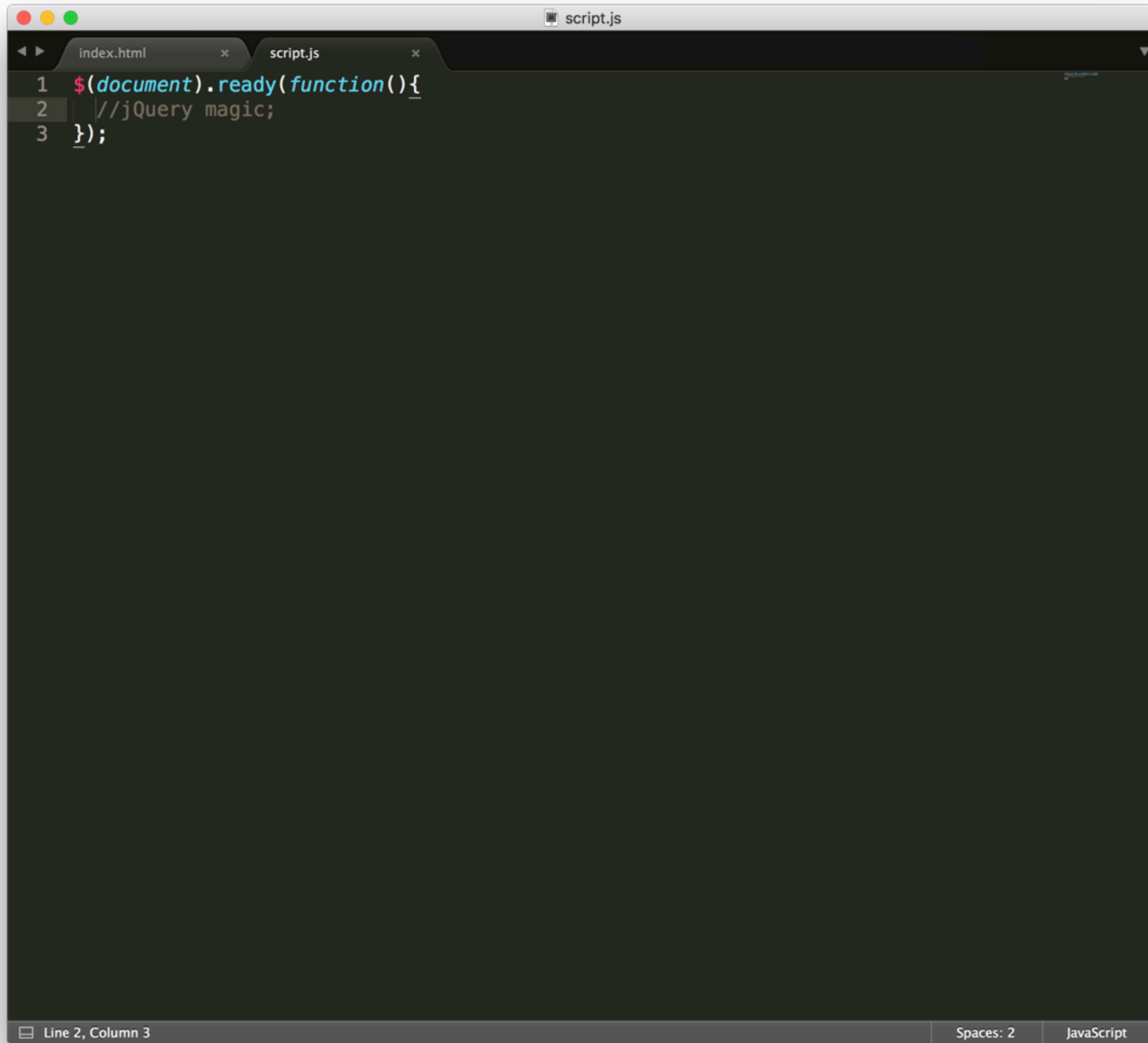
SWFObject

three.js

Web Font Loader

Troubleshooting

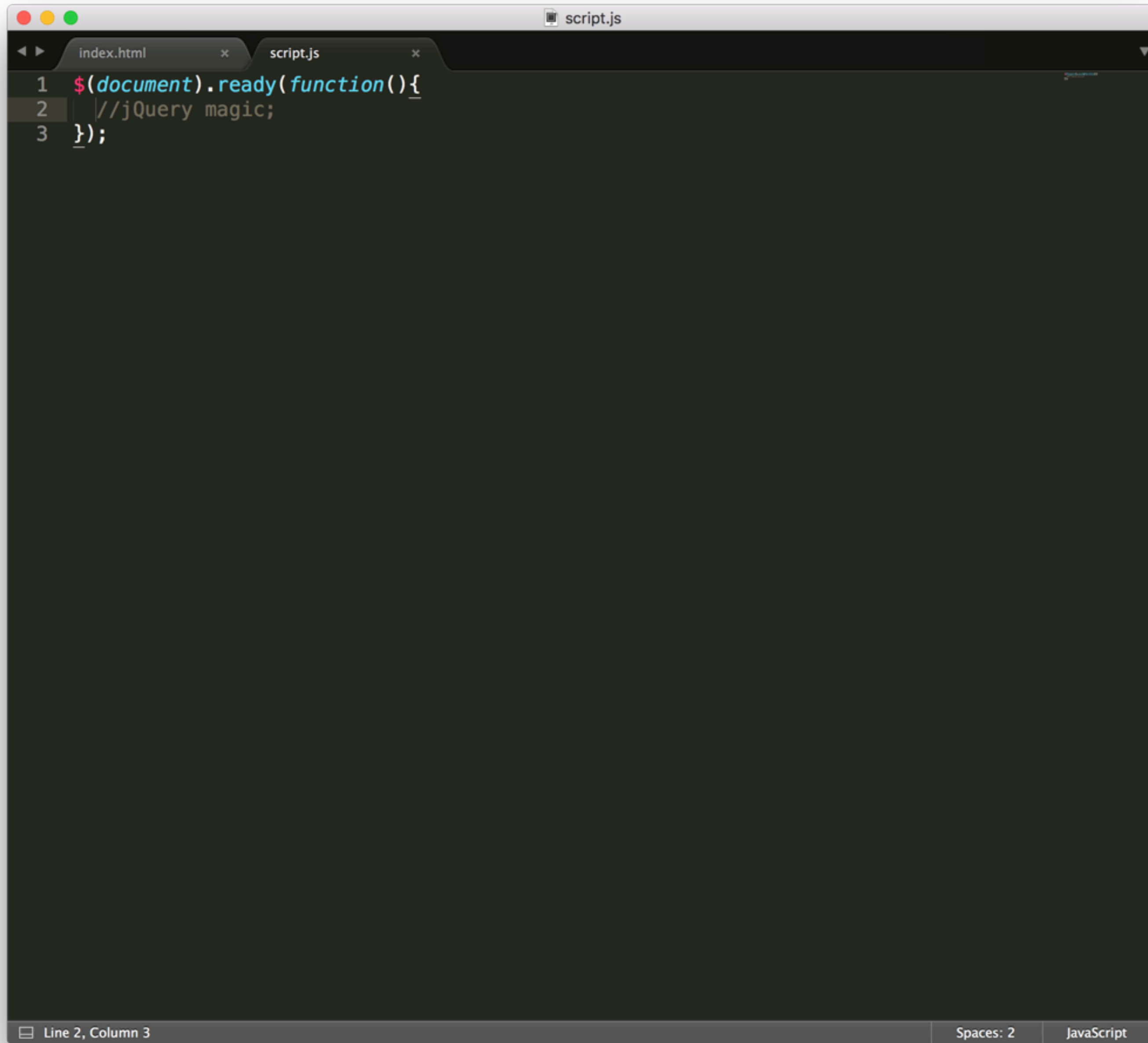
Now that we have loaded jQuery we can use it's selectors and functions in our other .js files.



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 'script.js'. The editor shows three lines of JavaScript code:
1 `$(document).ready(function(){`
2 `//jQuery magic;`
3 `});`
The code is syntax-highlighted: '\$' is red, 'document' is blue, 'ready' is green, 'function' is blue, and 'magic' is green. The status bar at the bottom shows 'Line 2, Column 3', 'Spaces: 2', and 'JavaScript'.

```
1 $(document).ready(function(){
2     //jQuery magic;
3 });
```

Let's break this down. `$()` says:
“Hey, jQuery things are about to happen.”



```
1 $(document).ready(function(){  
2   //jQuery magic;  
3 });
```

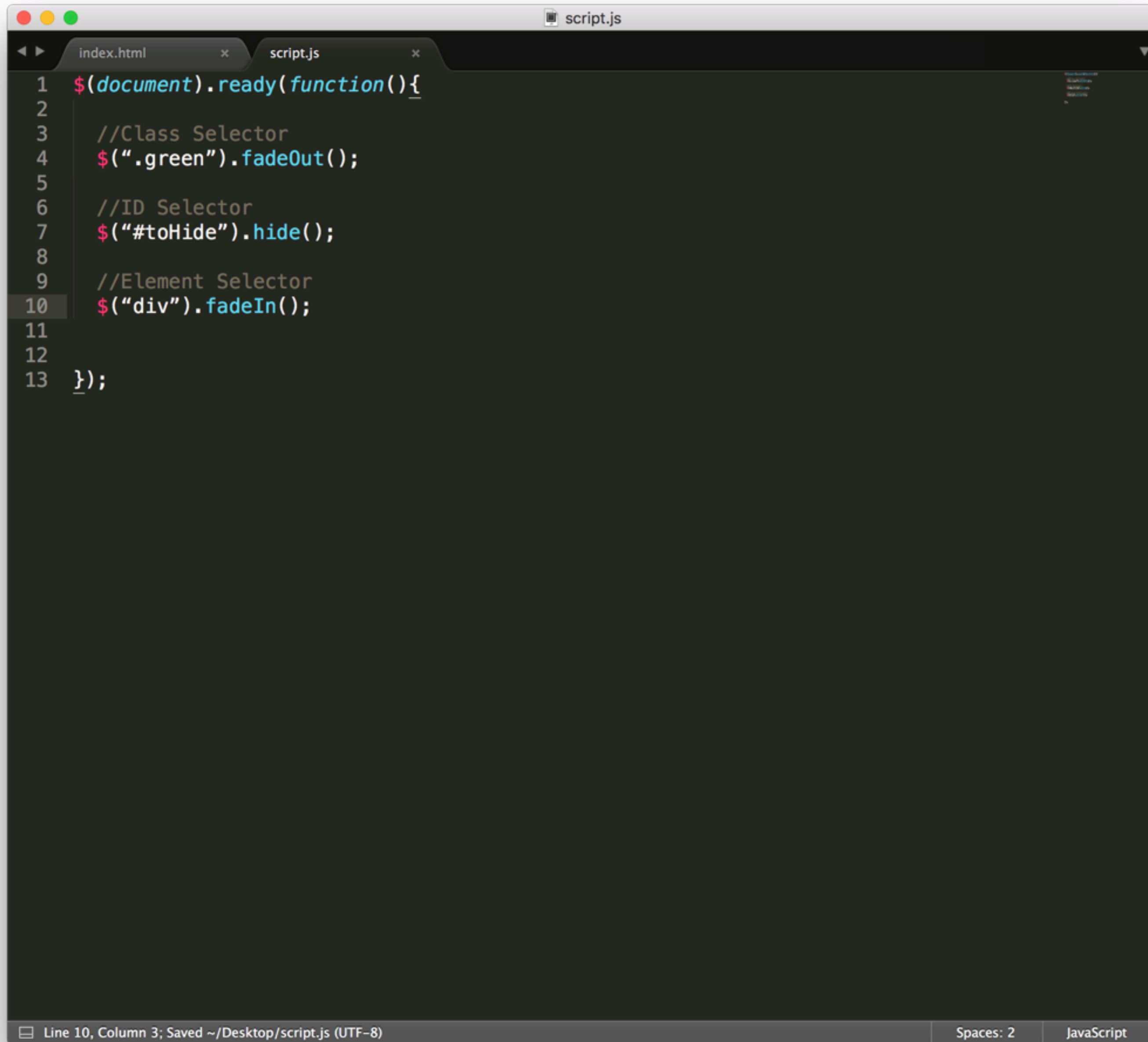
Line 2, Column 3 Spaces: 2 JavaScript

Next, we'll need to start up our jQuery magic using

```
$(document).ready();
```

Selectors

The first thing you'll want to do is select elements from the DOM to do things with. Selecting in jQuery works just like selecting in CSS.



The image shows a code editor window with two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, displaying the following JavaScript code:

```
1 $(document).ready(function(){  
2  
3     //Class Selector  
4     $(".green").fadeOut();  
5  
6     //ID Selector  
7     $("#toHide").hide();  
8  
9     //Element Selector  
10    $("div").fadeIn();  
11  
12  
13 });
```

The status bar at the bottom indicates 'Line 10, Column 3; Saved ~/Desktop/script.js (UTF-8)', 'Spaces: 2', and 'JavaScript'.

Selecting elements from the DOM in jQuery uses the below syntax:

`$("#target").someAction;`

**Next you'll want to do effects on your selections.
Here are some commonly used effects.**

Hide or show the matched elements:

`.hide();`
`.show();`

Display or hide the matched elements by fading them:

```
.fadeIn('fast');  
.fadeOut(1000);
```

Add a delay before the effect:

.delay(value);

Add a delay before the effect:

.delay(value);

Hide or show the matched elements:

`.toggle();`

Display or hide the matched elements by fading them:

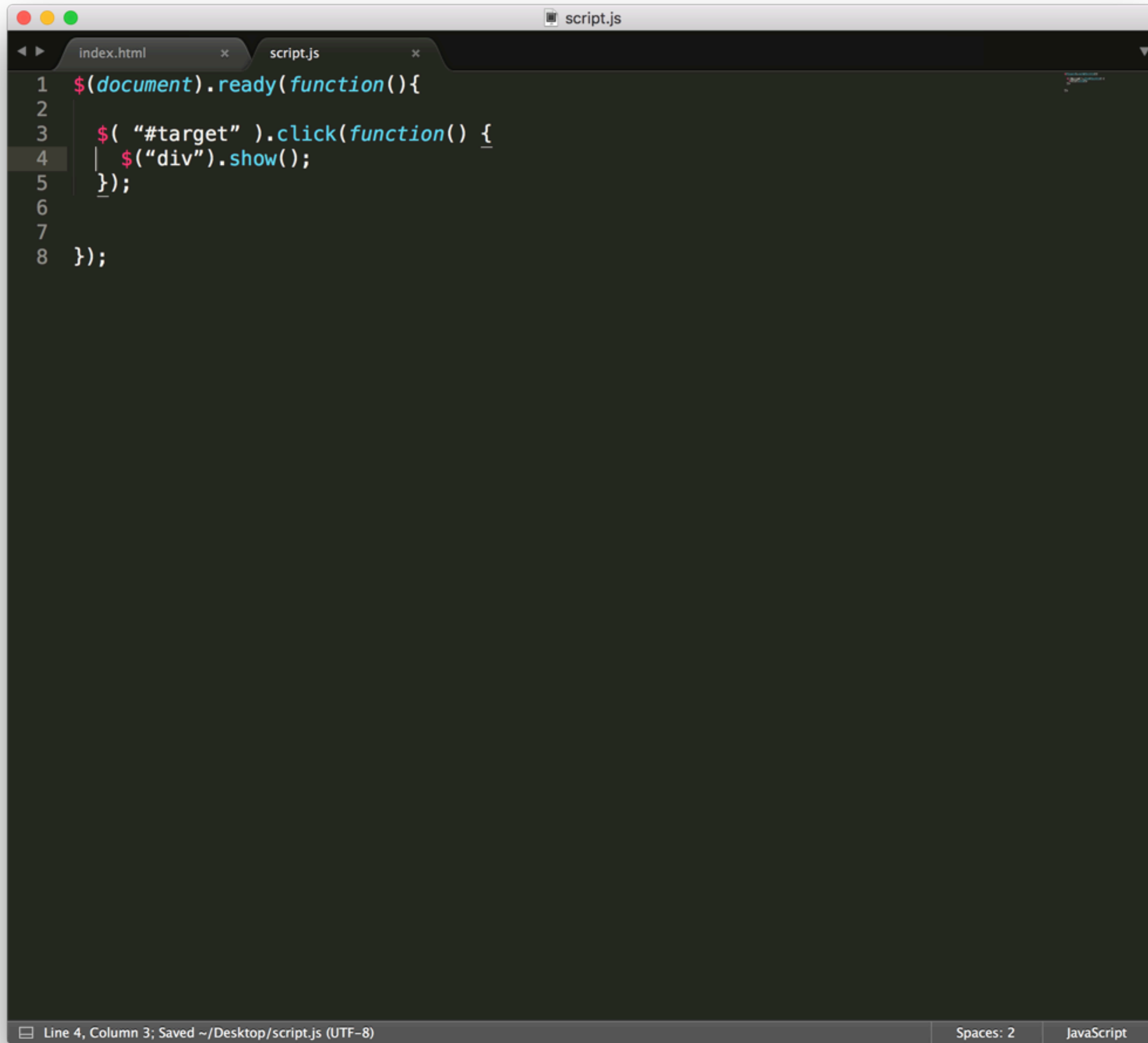
`.fadeToggle();`

js Fiddle Examples

You can also tie changes to events the user performs such as clicks, scrolls, etc.

```
$(‘thingToTouch’).event(function() {  
    $(‘thingToAffect’).effect();  
});
```

“Thing To Touch” is the HTML element you’ll click on, hover over, or otherwise interact with, and “thing to affect” is the HTML element that fades away, changes size, or undergoes some other transformation.

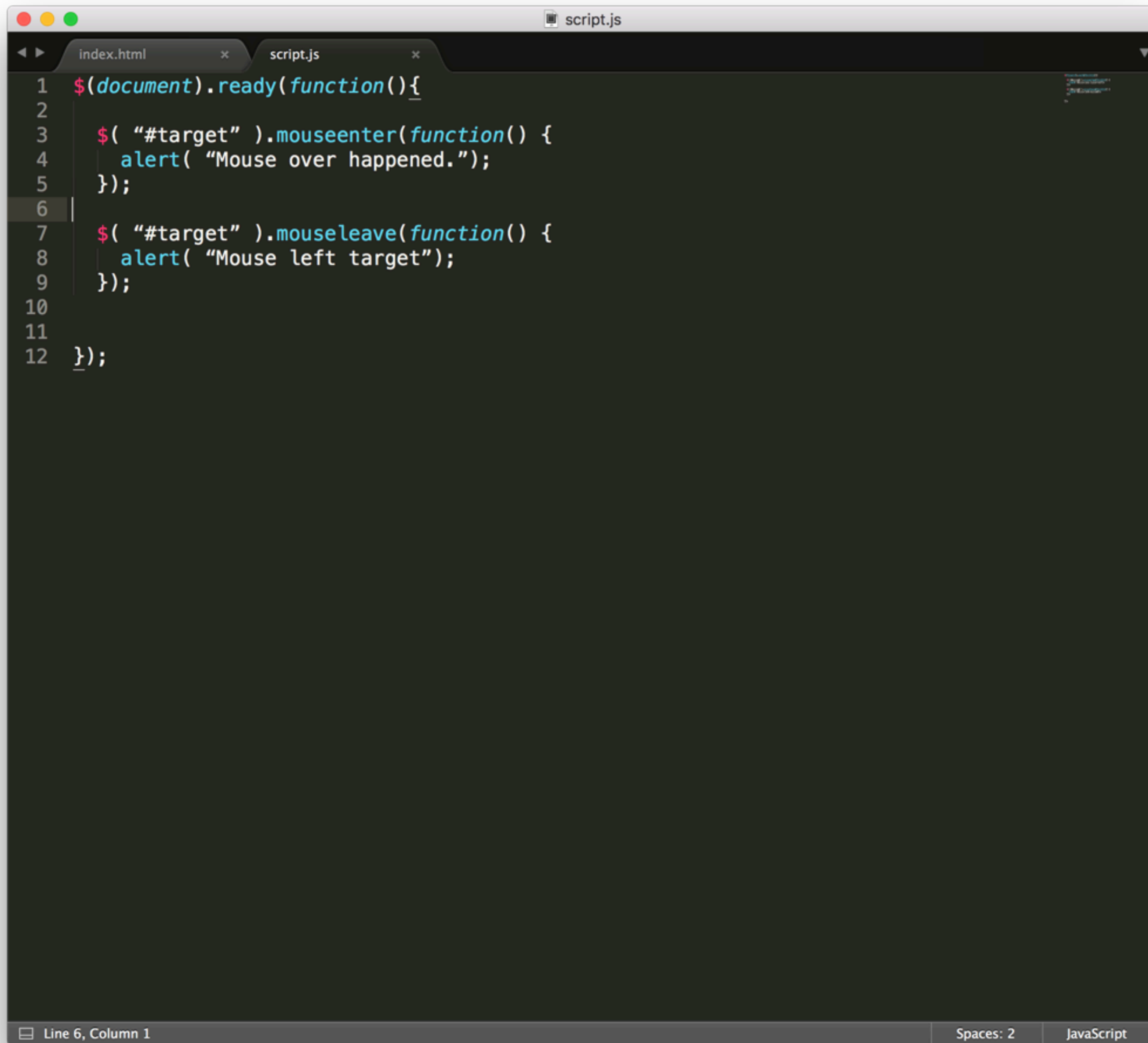


The image shows a code editor window with two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, displaying the following JavaScript code:

```
1 $(document).ready(function(){  
2  
3     $( "#target" ).click(function() {  
4         | $("div").show();  
5     });  
6  
7  
8 });
```

The code is written in a dark-themed editor. The first line is `$(document).ready(function(){`, followed by a blank line. The third line is `$("#target").click(function() {`. The fourth line is `| $("div").show();`, where the vertical bar indicates a cursor. The fifth line is `});`. There are two more blank lines, and the final line is `});`. The status bar at the bottom indicates 'Line 4, Column 3; Saved ~/Desktop/script.js (UTF-8)', 'Spaces: 2', and 'JavaScript'.

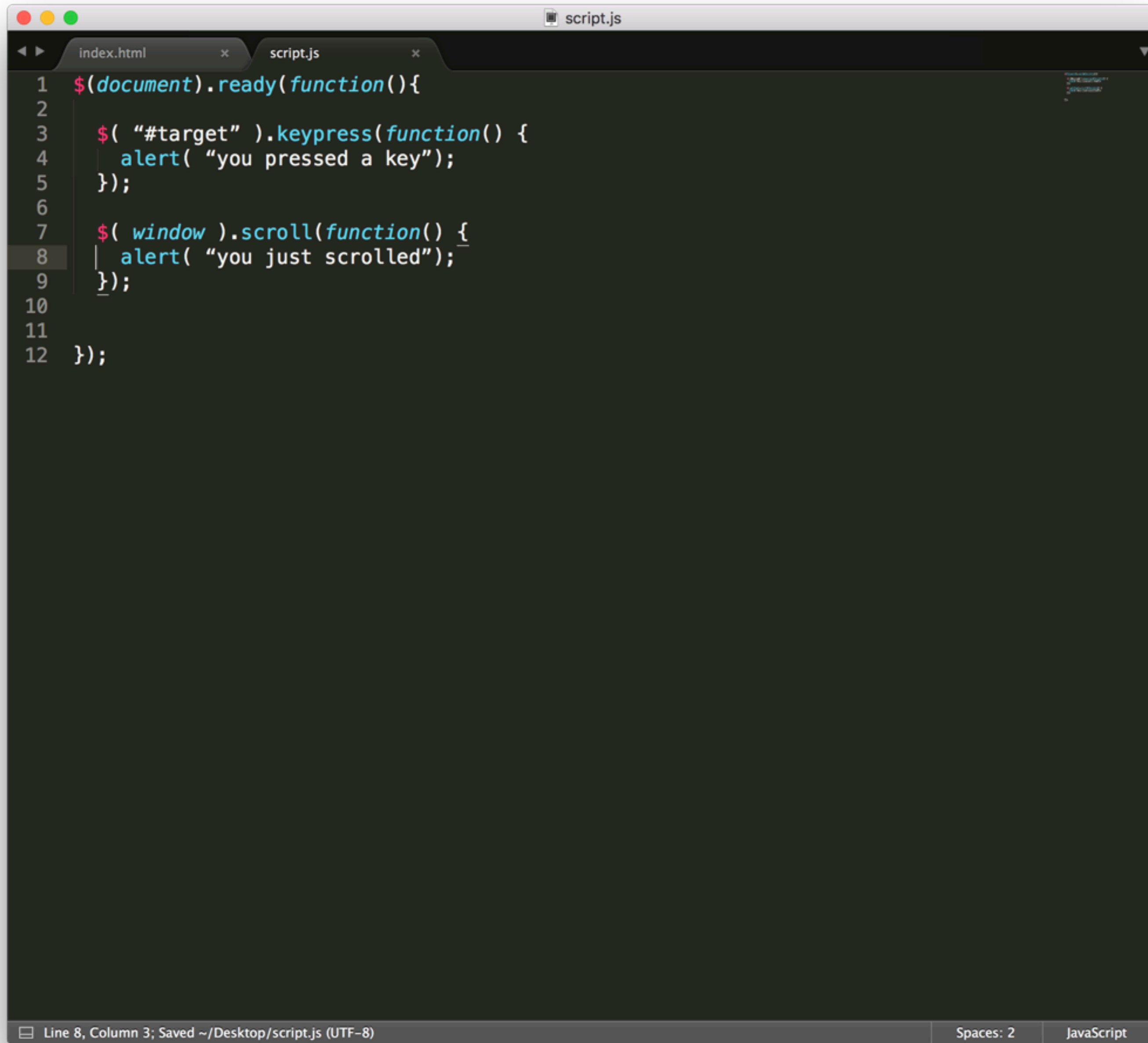
In this example when **#target** is clicked, all the divs on the page will fadeout.

A screenshot of a code editor window titled 'script.js'. The editor shows a JavaScript file with 12 lines of code. The code uses jQuery's \$.ready() method to attach mouseenter and mouseleave events to an element with the ID '#target'. The mouseenter event triggers an alert 'Mouse over happened.' and the mouseleave event triggers an alert 'Mouse left target'. The code is as follows:

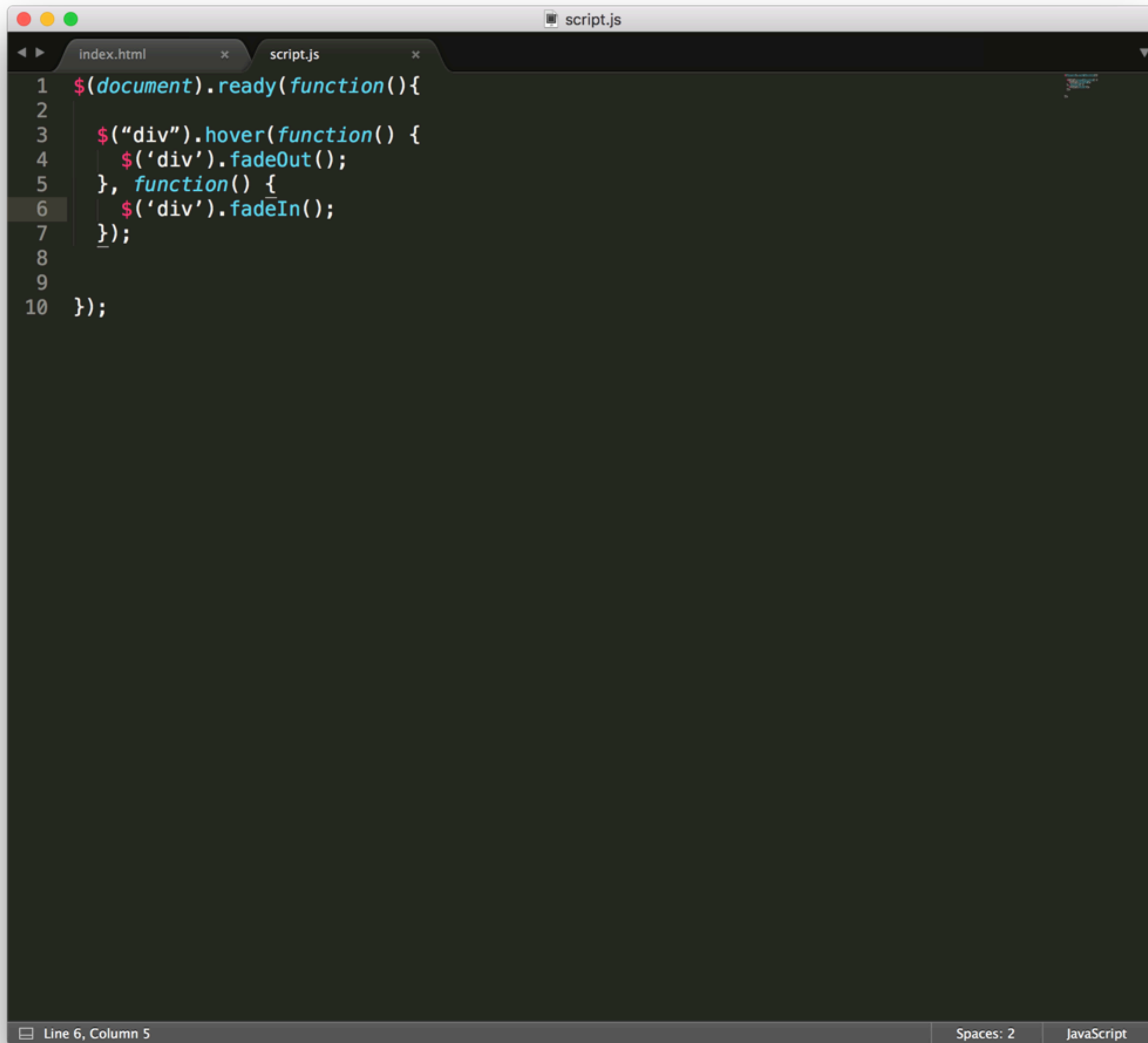
```
1 $(document).ready(function(){
2
3     $('#target').mouseenter(function() {
4         alert( "Mouse over happened." );
5     });
6
7     $('#target').mouseleave(function() {
8         alert( "Mouse left target" );
9     });
10
11
12 });
```

The editor interface includes a tab bar at the top with 'index.html' and 'script.js'. A status bar at the bottom indicates 'Line 6, Column 1', 'Spaces: 2', and 'JavaScript'.

In this example when the mouse enters the element with an id of #target and alert is triggered. Then when the mouse leaves another alert is triggered.

A screenshot of a code editor window titled 'script.js'. The editor has two tabs: 'index.html' and 'script.js'. The code is written in JavaScript and uses jQuery. It consists of 12 lines. Line 1: `$(document).ready(function(){`. Line 2: . Line 3: `$("#target").keypress(function() {`. Line 4: `alert("you pressed a key");`. Line 5: `});`. Line 6: . Line 7: `$(window).scroll(function() {`. Line 8: `alert("you just scrolled");`. Line 9: `});`. Line 10: . Line 11: . Line 12: `});`. The status bar at the bottom shows 'Line 8, Column 3; Saved ~/Desktop/script.js (UTF-8)', 'Spaces: 2', and 'JavaScript'.

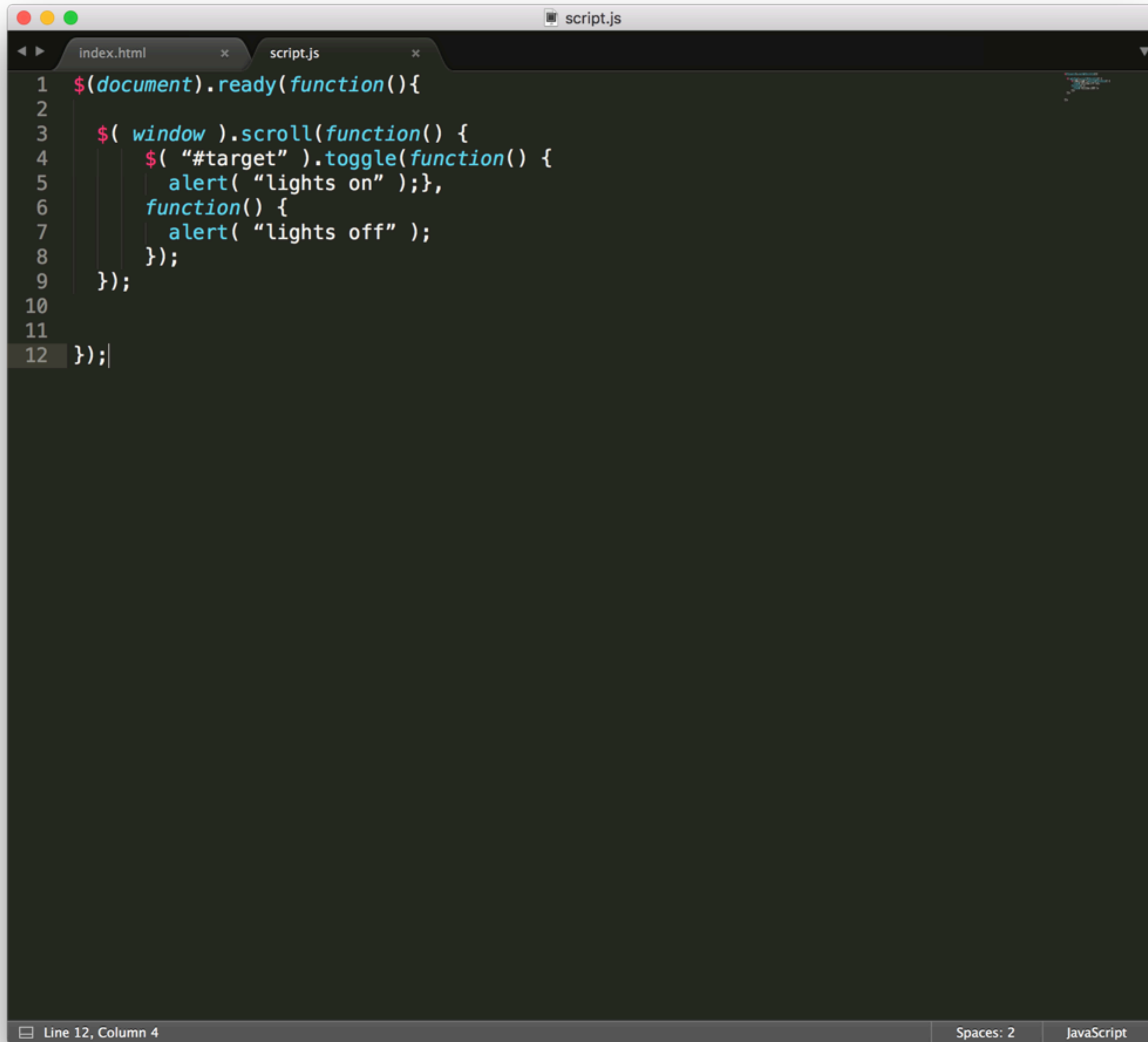
Here we see examples of the keypress event and the scroll event.



```
1 $(document).ready(function(){
2
3     $("div").hover(function() {
4         $('div').fadeOut();
5     }, function() {
6         $('div').fadeIn();
7     });
8
9
10 });
```

Line 6, Column 5 Spaces: 2 JavaScript

Here we see an example of the hover event which takes two functions. The first is for when you enter the target and the second is executed when your mouse leaves.

A screenshot of a code editor window with a dark theme. The window has two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, showing the following JavaScript code:

```
1 $(document).ready(function(){
2
3     $( window ).scroll(function() {
4         $( "#target" ).toggle(function() {
5             alert( "lights on" );},
6             function() {
7                 alert( "lights off" );
8             });
9     });
10
11
12 });|
```

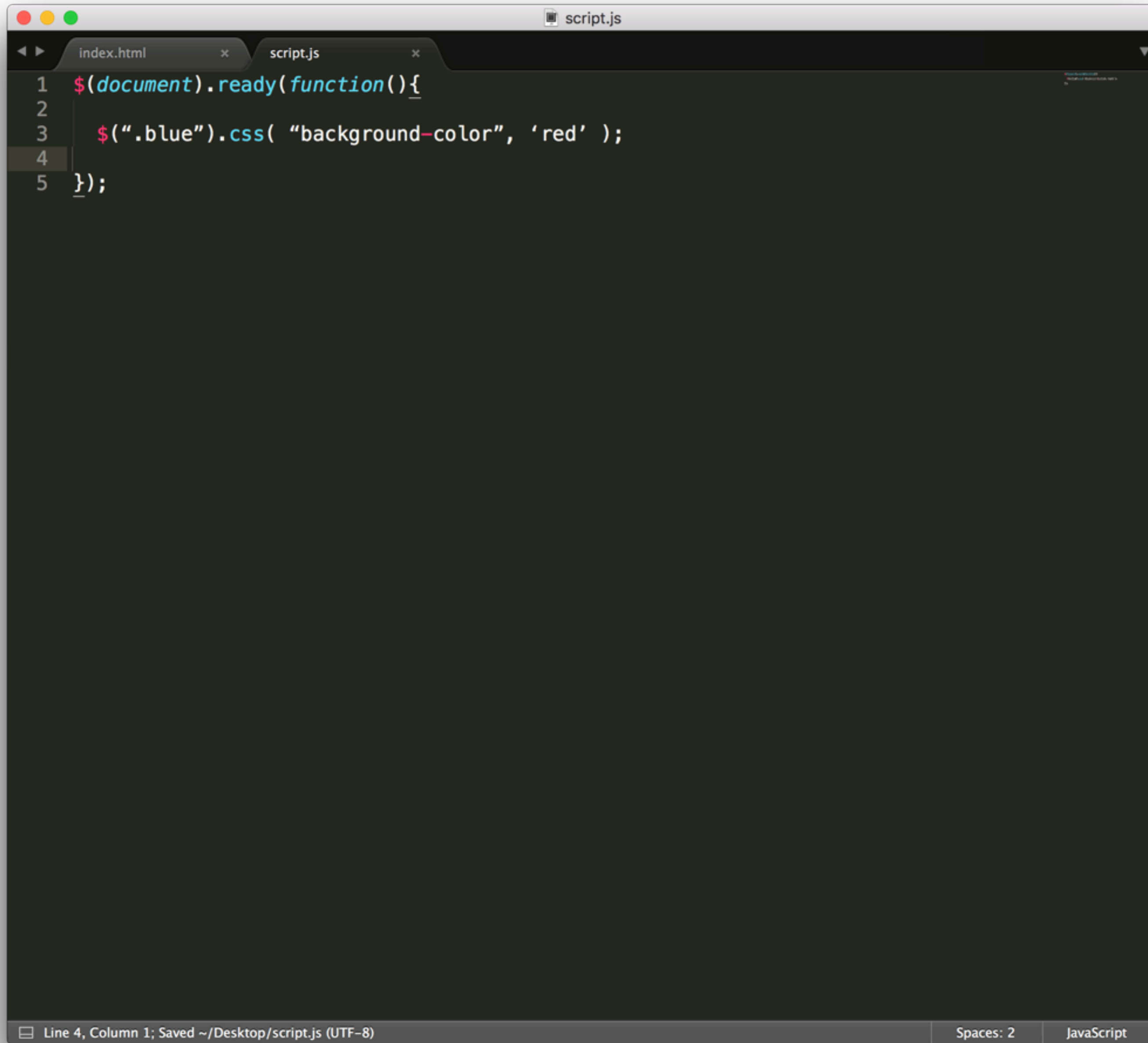
The code is syntactically highlighted. The status bar at the bottom indicates 'Line 12, Column 4', 'Spaces: 2', and 'JavaScript'.

This example combines the scroll event with a toggle event which combines two functions. The function will toggle between the two states.

js Fiddle Examples

You can also use jQuery to effect the css of your page. Here is the syntax for making css changes via jQuery:

```
$('#target').css( propertyName, value );
```

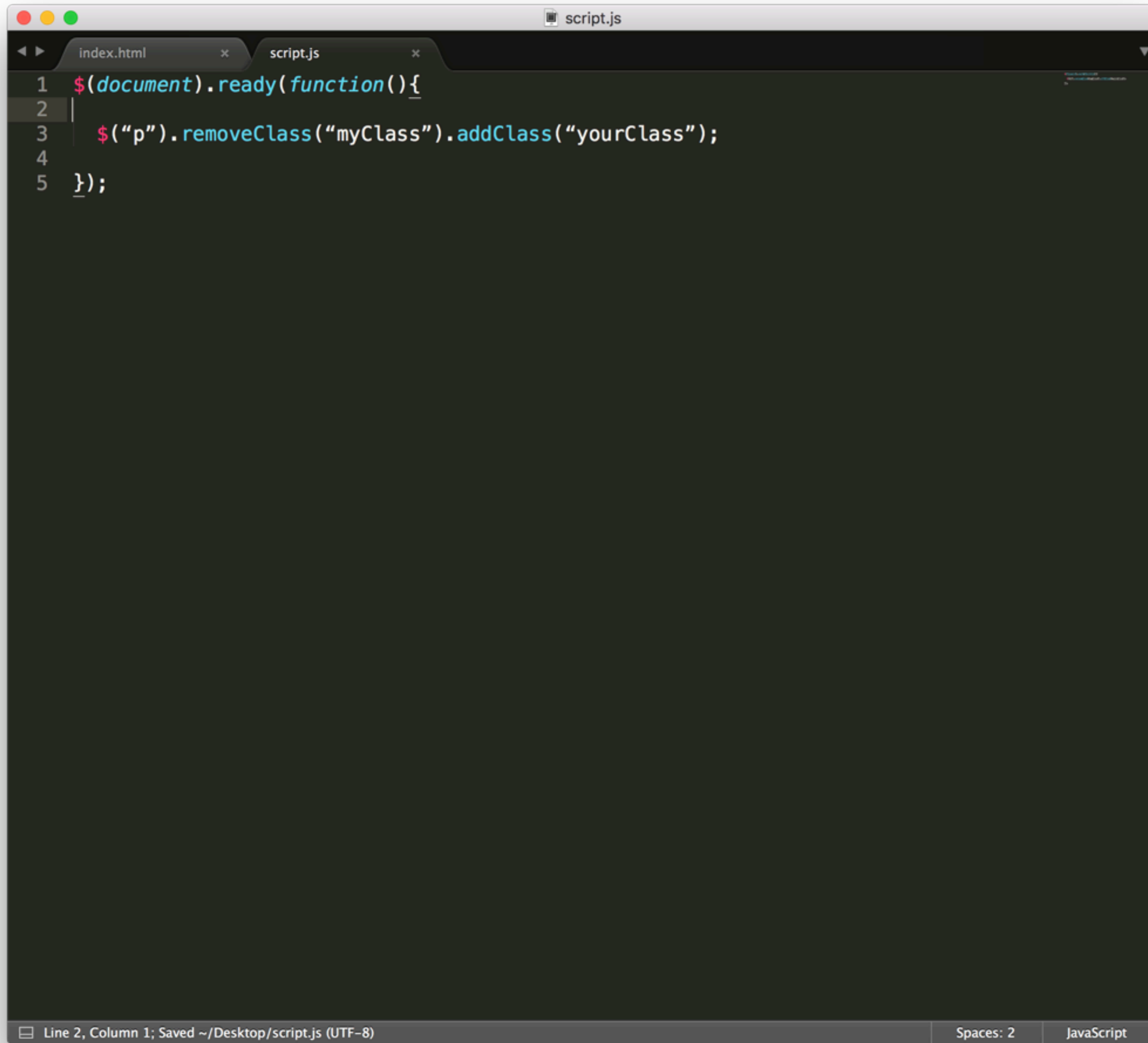


```
1 $(document).ready(function(){
2
3     $(".blue").css( "background-color", 'red' );
4
5 });
```

Line 4, Column 1; Saved ~/Desktop/script.js (UTF-8) Spaces: 2 JavaScript

Here is an example. Note however that you'll likely want to make this changes in relation to events on the page.

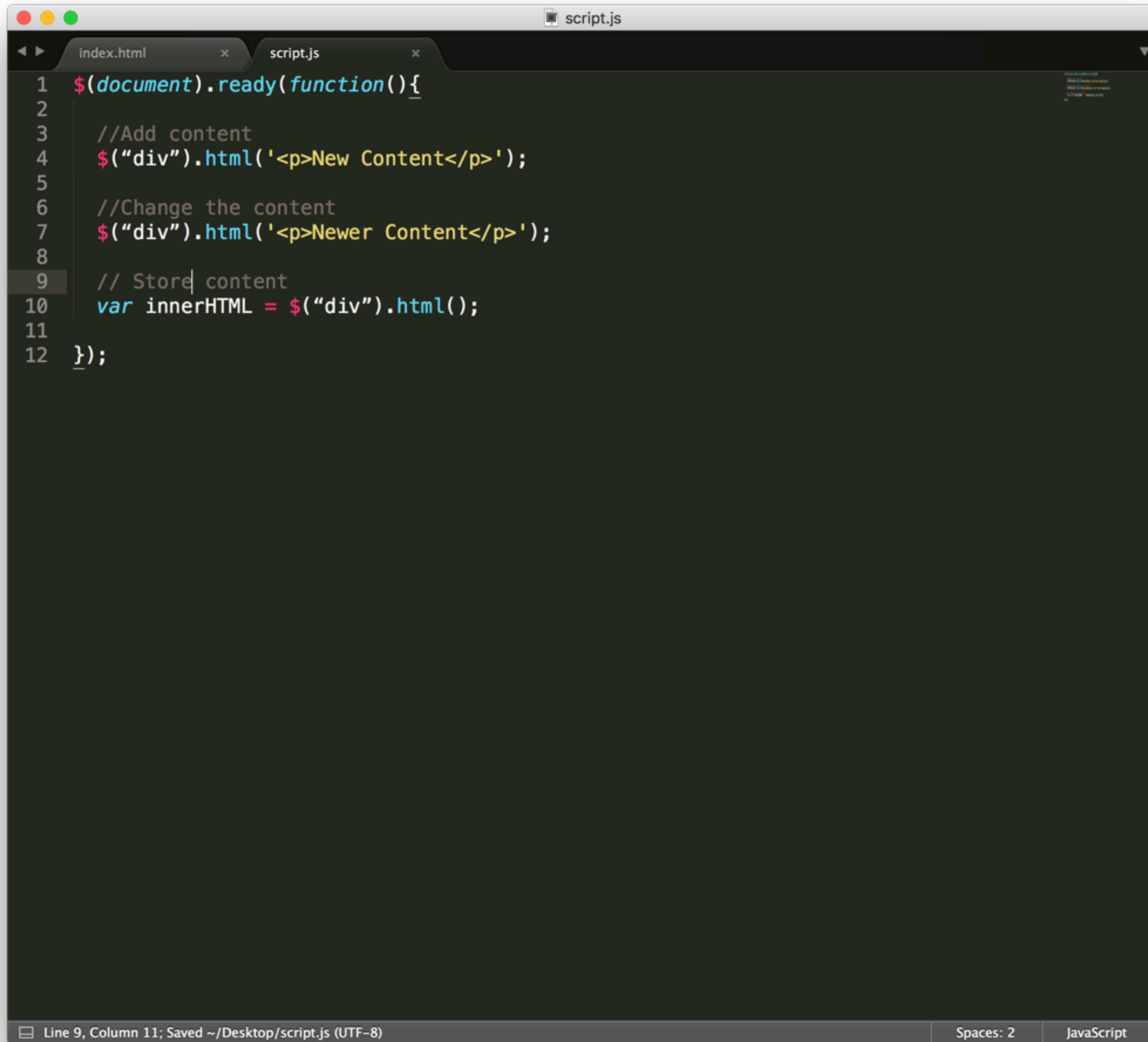
You can also use jQuery to change between CSS classes. This is a very common pattern and one of the best ways to make visual changes.

A screenshot of a code editor window with a dark theme. The window has a title bar with standard macOS window controls (red, yellow, green buttons) and a title 'script.js'. Below the title bar is a tab bar with two tabs: 'index.html' and 'script.js', with 'script.js' being the active tab. The main editing area shows five lines of JavaScript code. Line 1: `1 $(document).ready(function(){` Line 2: `2 |` (the cursor is on this line) Line 3: `3 $("p").removeClass("myClass").addClass("yourClass");` Line 4: `4` Line 5: `5 });` The bottom status bar shows 'Line 2, Column 1; Saved ~/Desktop/script.js (UTF-8)', 'Spaces: 2', and 'JavaScript'.

Note that you do not use a `!` to select the class. This example also shows that you can chain jQuery functions.

js Fiddle Examples

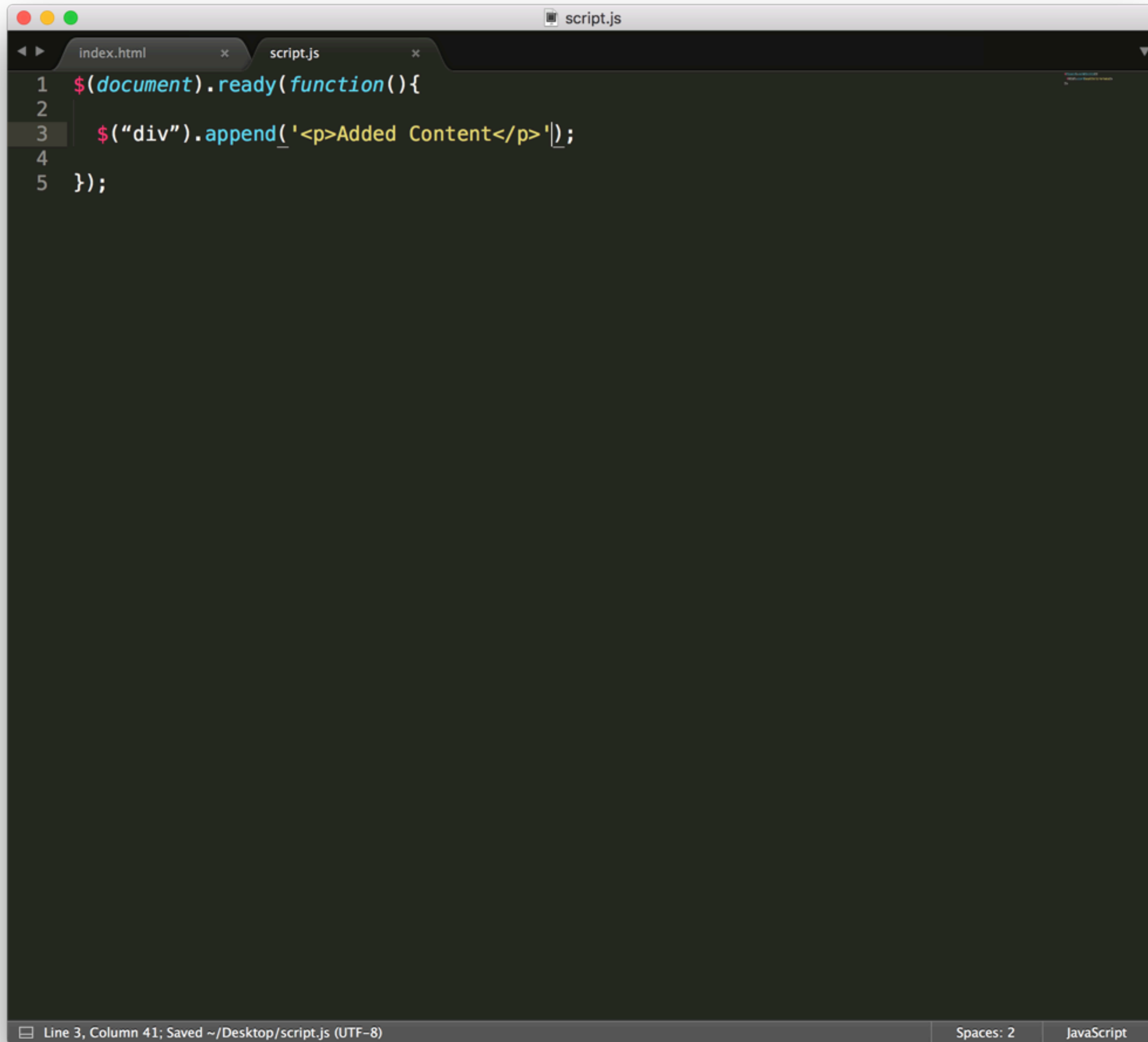
You can also use jQuery to edit the HTML of a page directly. In these examples you are adding or removing elements from the DOM.

A screenshot of a code editor window with a dark theme. The window has two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, showing a JavaScript file. The code is as follows:

```
1 $(document).ready(function(){
2
3     //Add content
4     $("div").html('<p>New Content</p>');
5
6     //Change the content
7     $("div").html('<p>Newer Content</p>');
8
9     // Store content
10    var innerHTML = $("div").html();
11
12 });
```

The code uses jQuery to manipulate the DOM. It first adds new content, then changes it, and finally stores the current content in a variable. The status bar at the bottom indicates 'Line 9, Column 11; Saved ~/Desktop/script.js (UTF-8)', 'Spaces: 2', and 'JavaScript'.

The **.html** function can be used to both set content or to get content from the DOM.

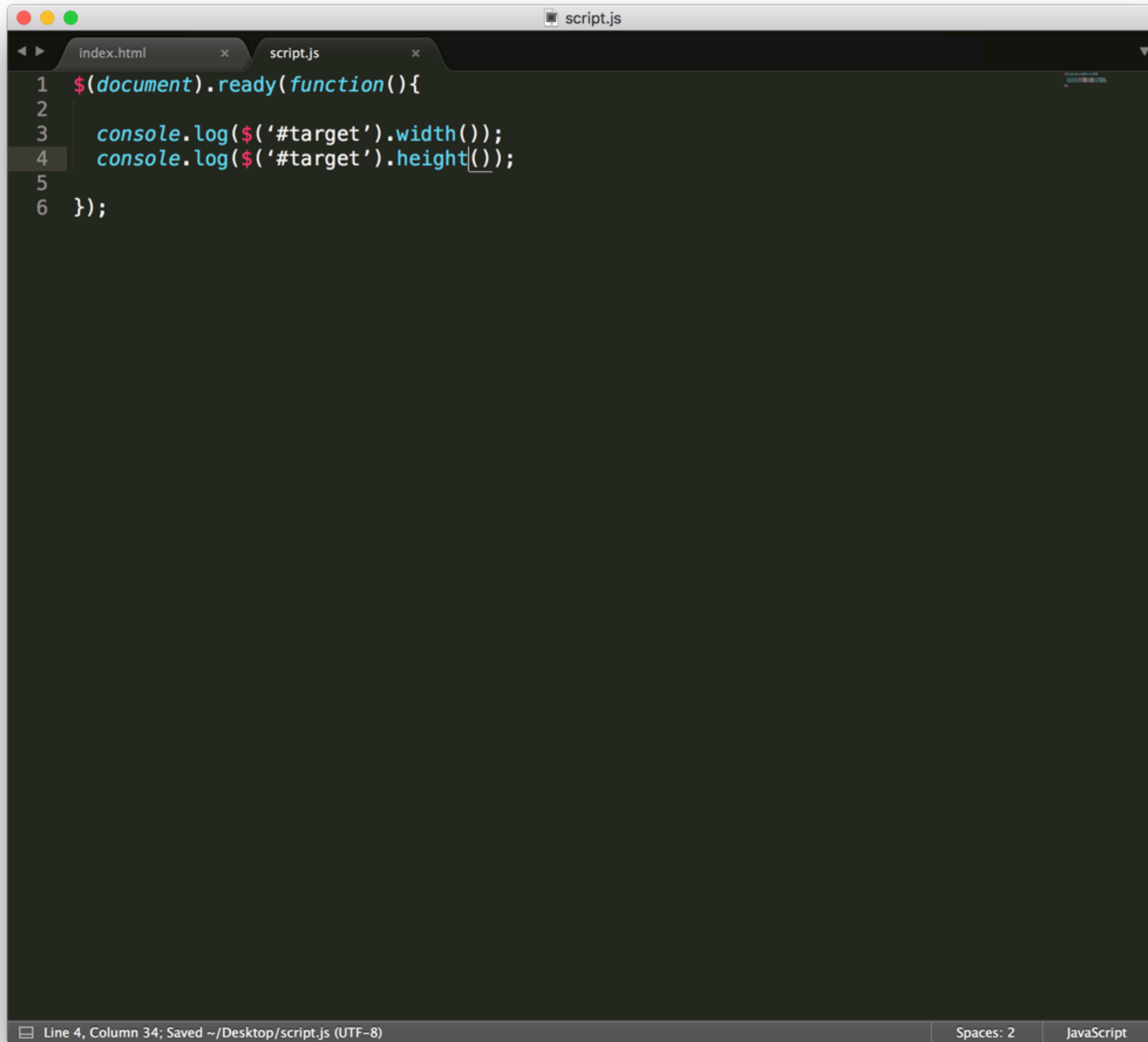


```
1 $(document).ready(function(){
2
3     $("div").append('<p>Added Content</p>');
4
5 });
```

Line 3, Column 41; Saved ~/Desktop/script.js (UTF-8) Spaces: 2 JavaScript

The **.append** function will add content at the end of the selected element.

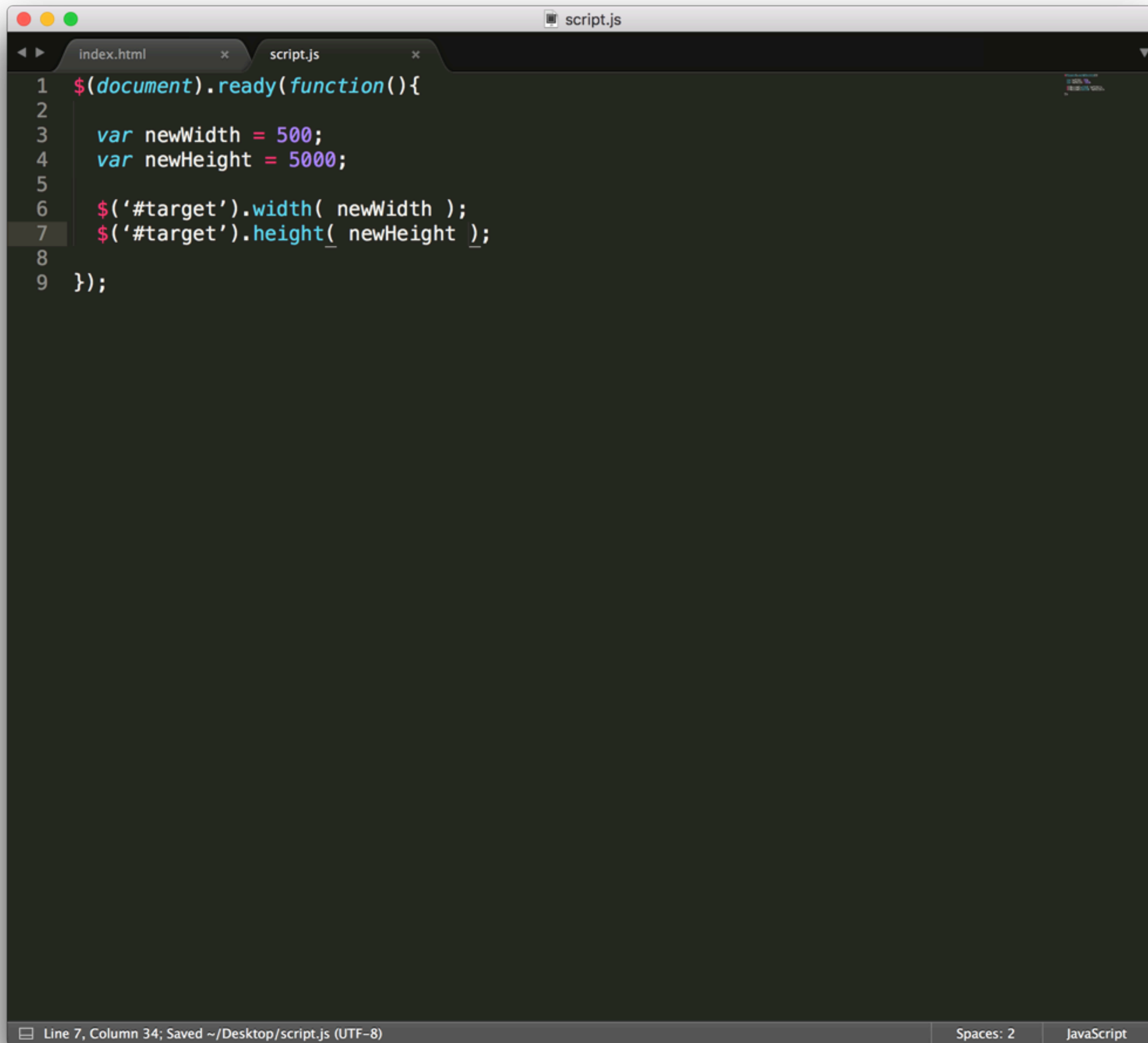
Finally you can use jQuery to return information about the DOM.

A screenshot of a code editor window with a dark theme. The window has a title bar with red, yellow, and green window control buttons on the left and a tab labeled 'script.js' on the right. The editor shows a JavaScript file with the following code:

```
1 $(document).ready(function(){
2
3   console.log($('#target').width());
4   console.log($('#target').height());
5
6 });
```

The cursor is positioned at the end of line 4. The status bar at the bottom indicates 'Line 4, Column 34; Saved ~/Desktop/script.js (UTF-8)', 'Spaces: 2', and 'JavaScript'.

In this example we're getting the width and height of the element with an id of target using **.width** and **.height**. We then print the values to the console.



```
1 $(document).ready(function(){
2
3     var newWidth = 500;
4     var newHeight = 5000;
5
6     $('#target').width( newWidth );
7     $('#target').height( newHeight );
8
9 });
```

The image shows a code editor window with two tabs: 'index.html' and 'script.js'. The 'script.js' tab is active, displaying the above JavaScript code. The code uses jQuery's `$(document).ready()` to execute a function that sets the width and height of an element with the ID 'target'. The width is set to 500 and the height to 5000. The editor has a dark theme and a status bar at the bottom indicating 'Line 7, Column 34; Saved ~/Desktop/script.js (UTF-8)', 'Spaces: 2', and 'JavaScript'.

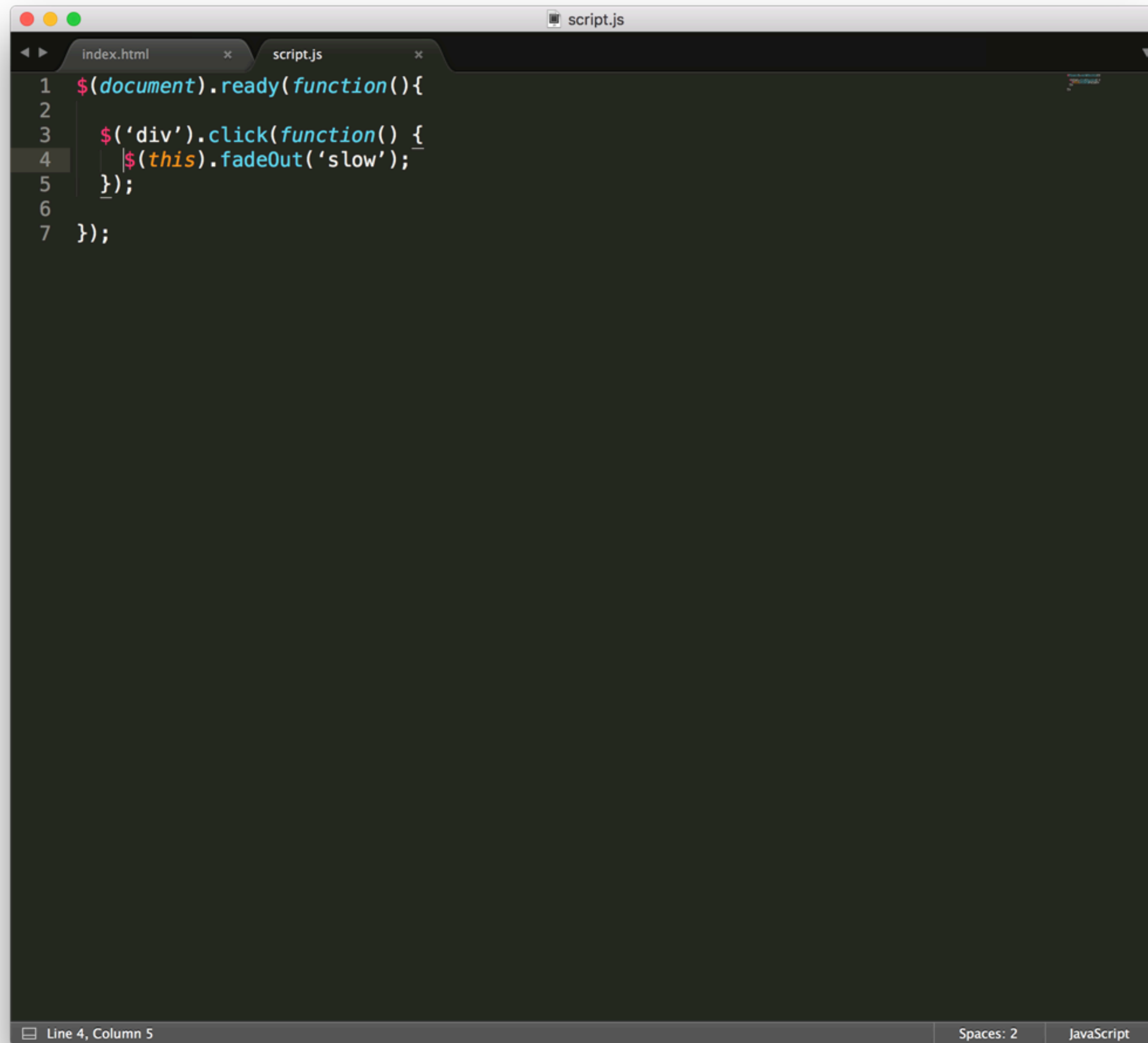
You can also use **.width** and **.height** to set these attributes on an element or group of elements.

js Fiddle Examples

'this' is Important!

The **this** keyword refers to the jQuery object you're currently doing something with.

`$(this)`, and the event will only affect the element you're currently doing something with (for example, clicking on or mousing over).



```
1 $(document).ready(function(){
2
3     $('div').click(function() {
4         $(this).fadeOut('slow');
5     });
6
7 });
```

Line 4, Column 5

Spaces: 2

JavaScript