JavaScript for Beginners

Building Interactive Web Apps on the Frontend

Brought to you by Dev Bootcamp

http://bit.ly/javascript-DBC-slides

Hi everyone!

Some "rules" for today:

- We are here to support your learning!
- Ask **every single question** you have.
- Take advantage of the amazing person next to you!
- If you want to do it, **do it**. Have fun with this.

Welcome!

Let's get to know each other!

- What's your name, and who are you?
- What do you want to get out of this session?
- What's **something quirky** about you?

Outline

Overview

- Why JavaScript matters
- Tools for writing/learning JavaScript
- What you can build with Javascript

Coding Basics

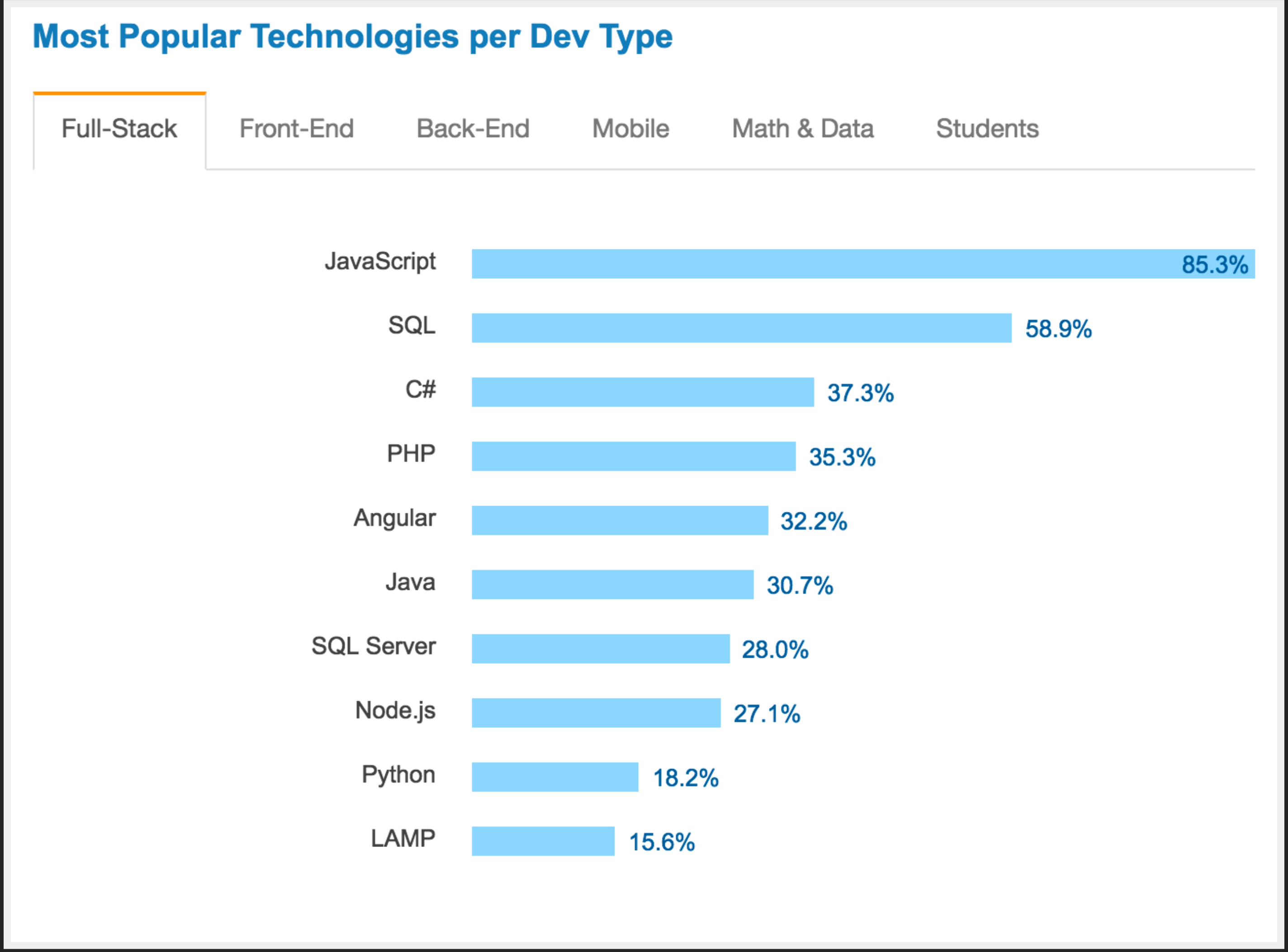
- Data structures for storing
- Loops, conditionals and functions
- Breakout - use your tools!

jQuery

- Use jQuery to change a webpage based on user interaction
- Breakout - use your tools!

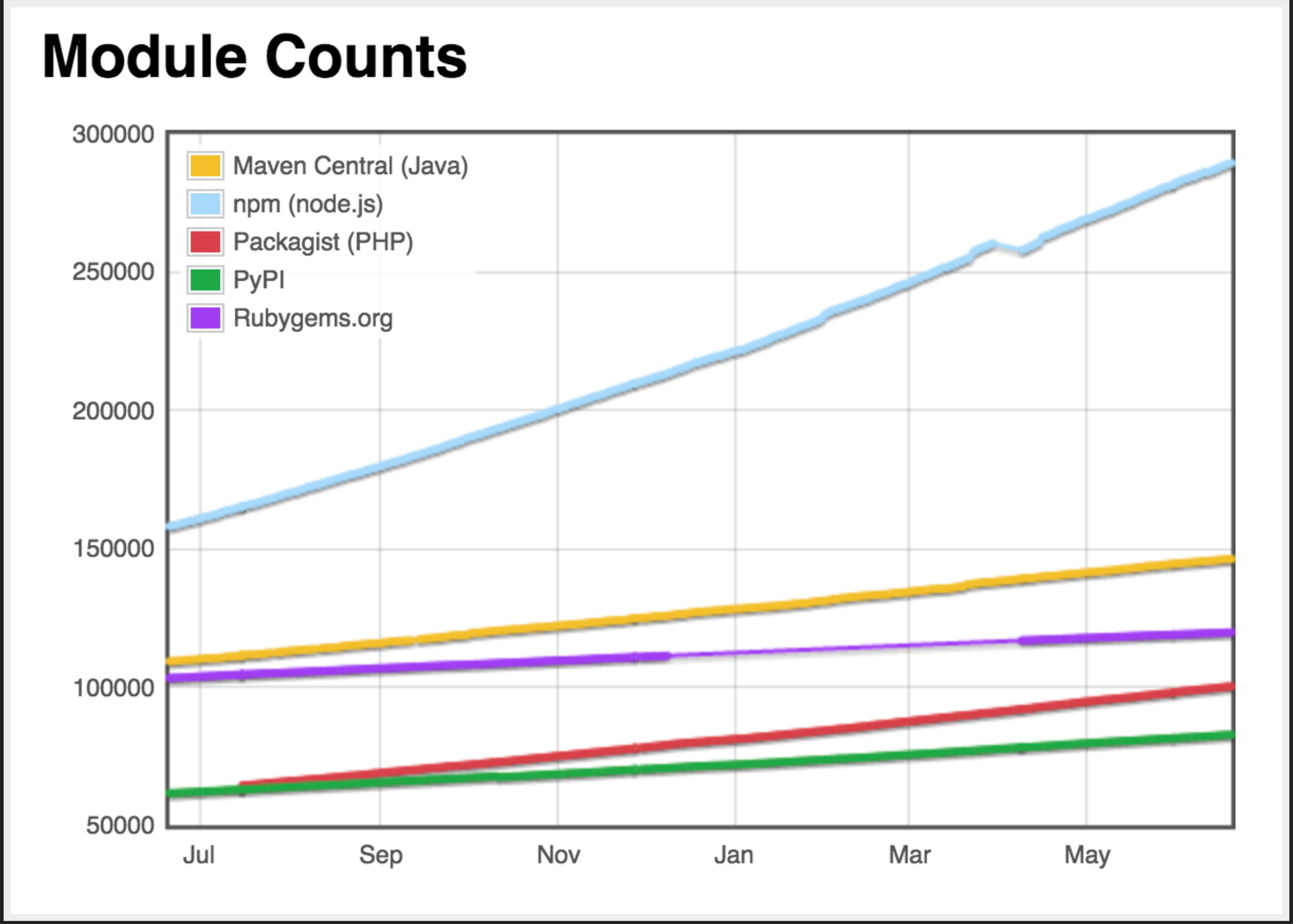
Why does JavaScript matter?

Javascript is the **most commonly used** programming language on earth.



Stack Overflow Developer Survey 2016

And it's growing at the fastest rate.



modulecounts.com

Which companies use Javascript?

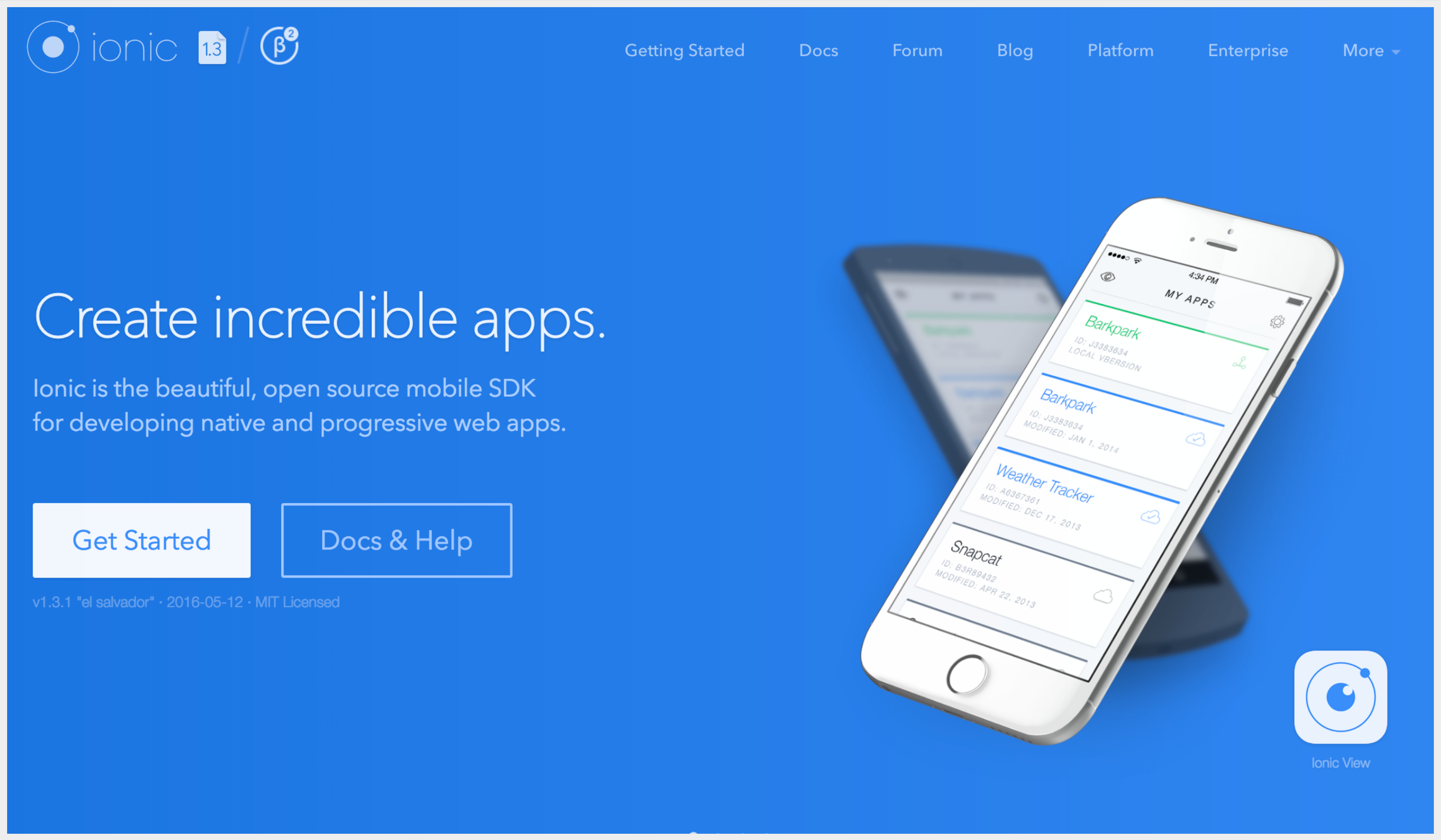
Every single one.

What can you build with Javascript?

Anything.

Interactive Web Apps

Mobile Apps



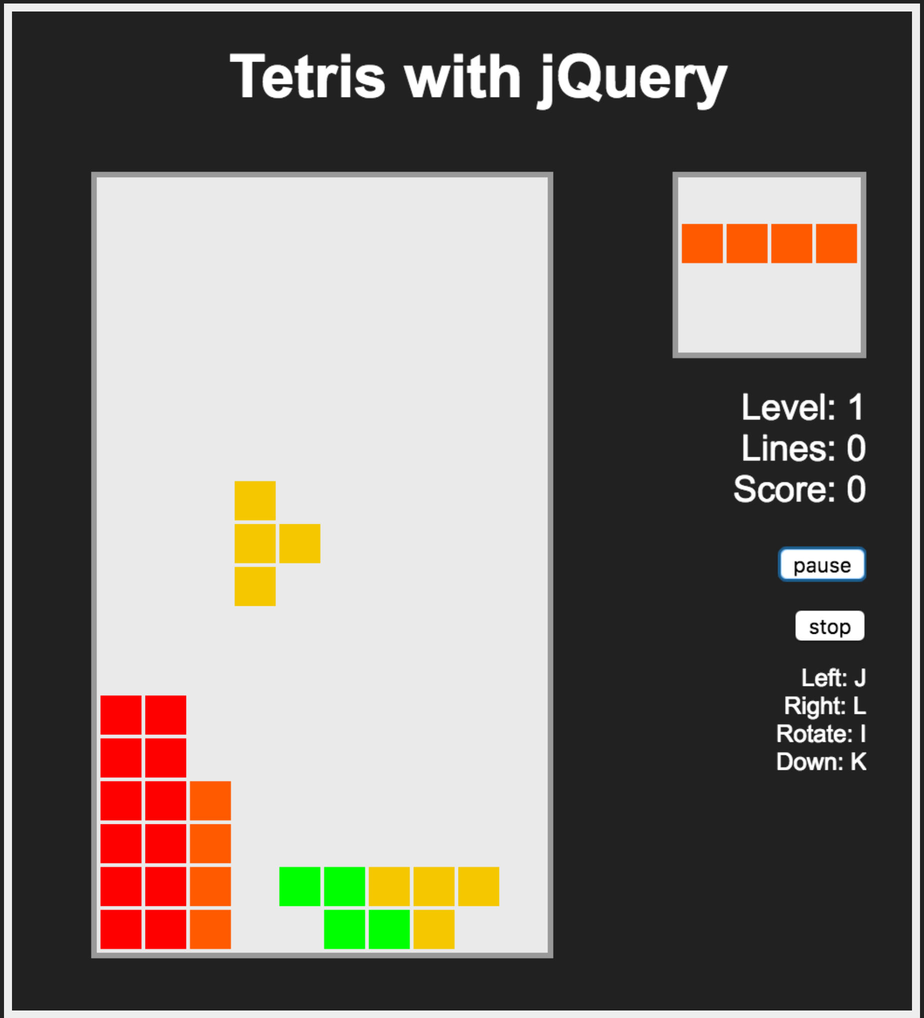
Create incredible apps.

Ionic is the beautiful, open source mobile SDK
for developing native and progressive web apps.

v1.3.1 "el salvador" · 2016-05-12 · MIT Licensed

http://ionicframework.com

Games



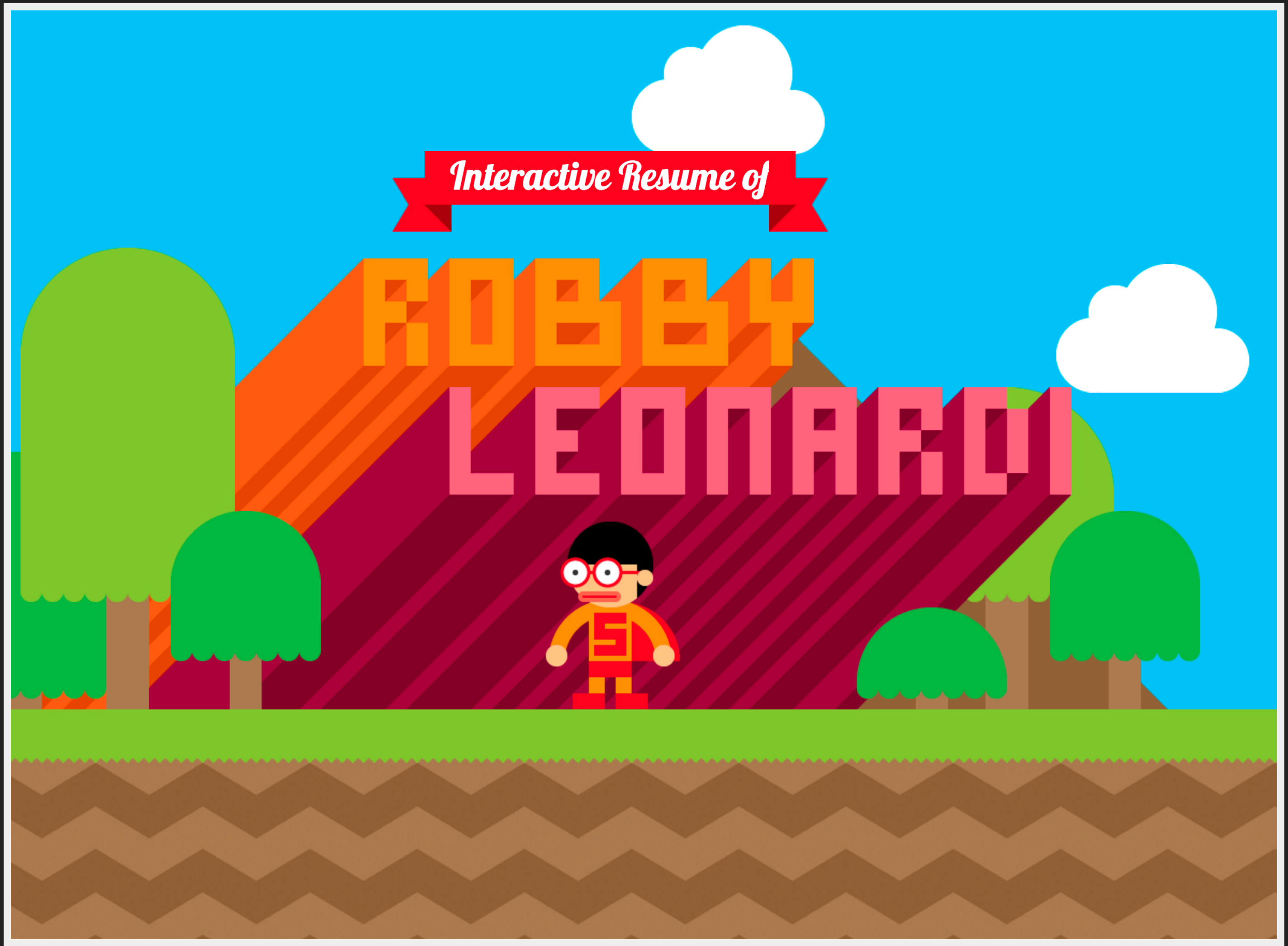
Tetris with jQuery

Level: 1
Lines: 0
Score: 0

Left: J
Right: L
Rotate: I
Down: K

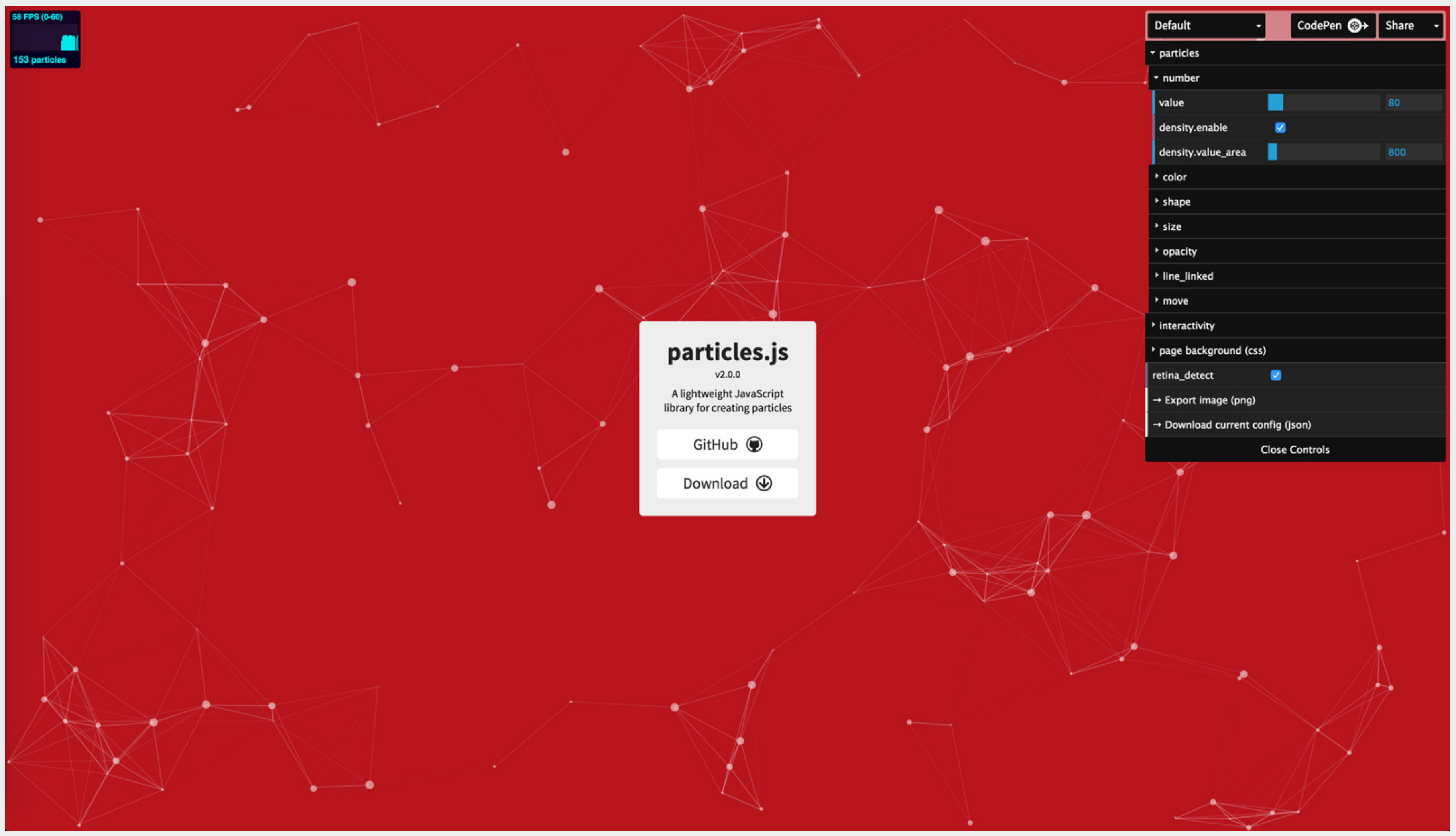
http://fmarcia.info/jquery/tetris/tetris.html

Interactive Resumes



http://rleonardi.com/interactive-resume/

Other Cool Stuff



particles.js

v2.0.0

A lightweight JavaScript library for creating particles

value 80

density.enable ☑

density.value_area 800

retina_detect ☑

http://vincentgarreau.com/particles.js/

JavaScript tools

Google Chrome

A web browser with great JavaScript development tools.

Sublime Text

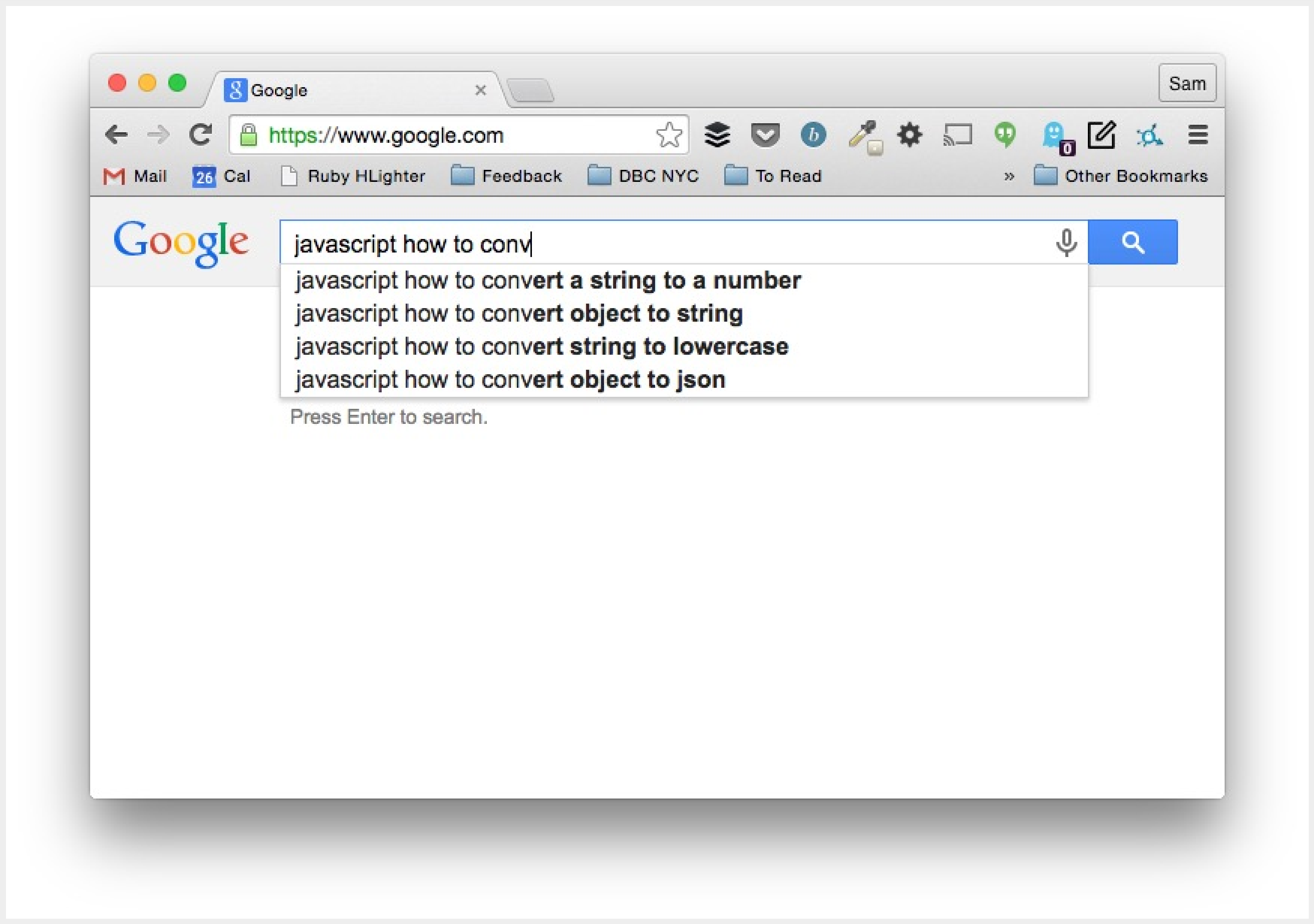
A powerful but user-friendly text editor.

jQuery

A JavaScript library that makes it easier for us to interact with webpages.

Research techniques

Google search and autocomplete will quickly become your best friend



javascript how to conv

javascript how to conv**ert a string to a number**

javascript how to conv**ert object to string**

javascript how to conv**ert string to lowercase**

javascript how to conv**ert object to json**

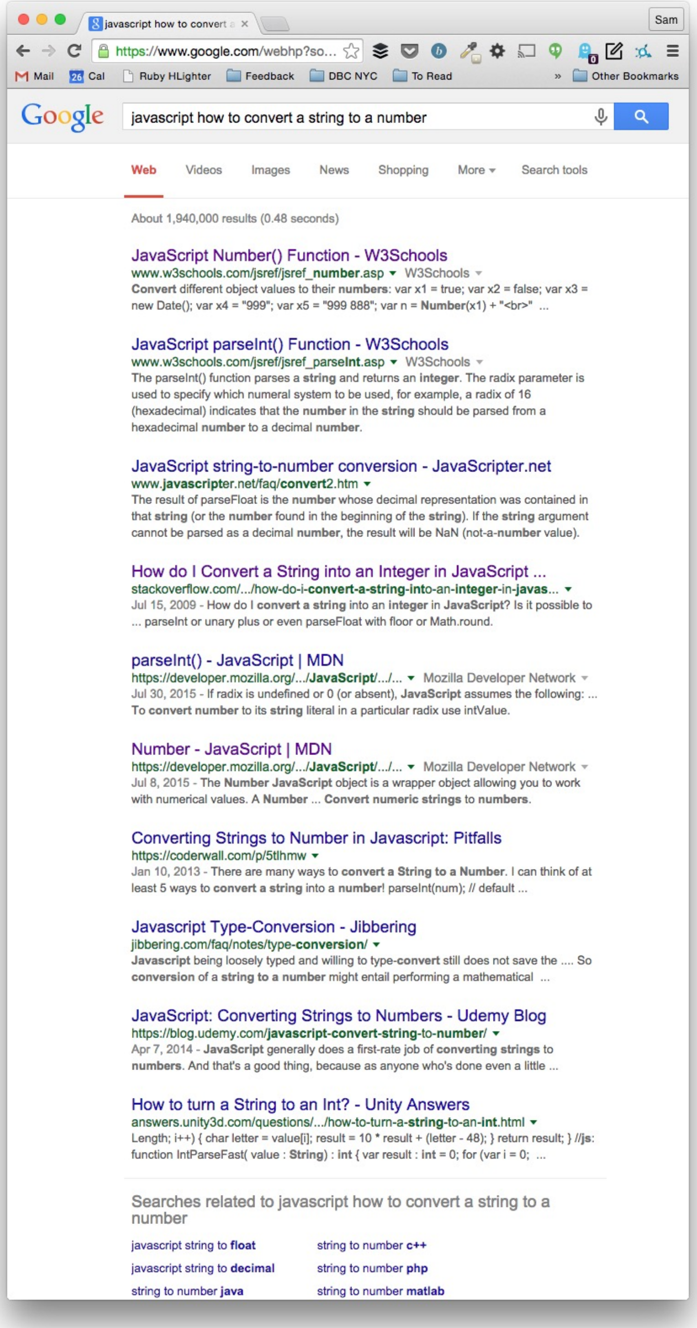
Press Enter to search.

Research techniques

Your ultimate goal is a super fast **feedback loop**.

Research techniques

Scan through the Google search results and work out which look most relevant to your issue.



javascript how to convert a string to a number

About 1,940,000 results (0.48 seconds)

JavaScript Number() Function - W3Schools
www.w3schools.com/jsref/jsref_number.asp ▾ W3Schools ▾
Convert different object values to their numbers: var x1 = true; var x2 = false; var x3 = new Date(); var x4 = "999"; var x5 = "999 888"; var n = Number(x1) + "<br>" ...

JavaScript parseInt() Function - W3Schools
www.w3schools.com/jsref/jsref_parseInt.asp ▾ W3Schools ▾
The parseInt() function parses a string and returns an integer. The radix parameter is used to specify which numeral system to be used, for example, a radix of 16 (hexadecimal) indicates that the number in the string should be parsed from a hexadecimal number to a decimal number.

JavaScript string-to-number conversion - JavaScripter.net
www.javascripter.net/faq/convert2.htm ▾
The result of parseFloat is the number whose decimal representation was contained in that string (or the number found in the beginning of the string). If the string argument cannot be parsed as a decimal number, the result will be NaN (not-a-number value).

How do I Convert a String into an Integer in JavaScript ...
stackoverflow.com/.../how-do-i-convert-a-string-into-an-integer-in-javas... ▾
Jul 15, 2009 - How do I convert a string into an integer in JavaScript? Is it possible to ... parseInt or unary plus or even parseFloat with floor or Math.round.

parseInt() - JavaScript | MDN
https://developer.mozilla.org/.../JavaScript/.../... ▾ Mozilla Developer Network ▾
Jul 30, 2015 - If radix is undefined or 0 (or absent), JavaScript assumes the following: ... To convert number to its string literal in a particular radix use intValue.

Number - JavaScript | MDN
https://developer.mozilla.org/.../JavaScript/.../... ▾ Mozilla Developer Network ▾
Jul 8, 2015 - The Number JavaScript object is a wrapper object allowing you to work with numerical values. A Number ... Convert numeric strings to numbers.

Converting Strings to Number in Javascript: Pitfalls
https://coderwall.com/p/5tlhmw ▾
Jan 10, 2013 - There are many ways to convert a String to a Number. I can think of at least 5 ways to convert a string into a number! parseInt(num); // default ...

Javascript Type-Conversion - Jibbering
jibbering.com/faq/notes/type-conversion/ ▾
Javascript being loosely typed and willing to type-convert still does not save the .... So conversion of a string to a number might entail performing a mathematical ...

JavaScript: Converting Strings to Numbers - Udemy Blog
https://blog.udemy.com/javascript-convert-string-to-number/ ▾
Apr 7, 2014 - JavaScript generally does a first-rate job of converting strings to numbers. And that's a good thing, because as anyone who's done even a little ...

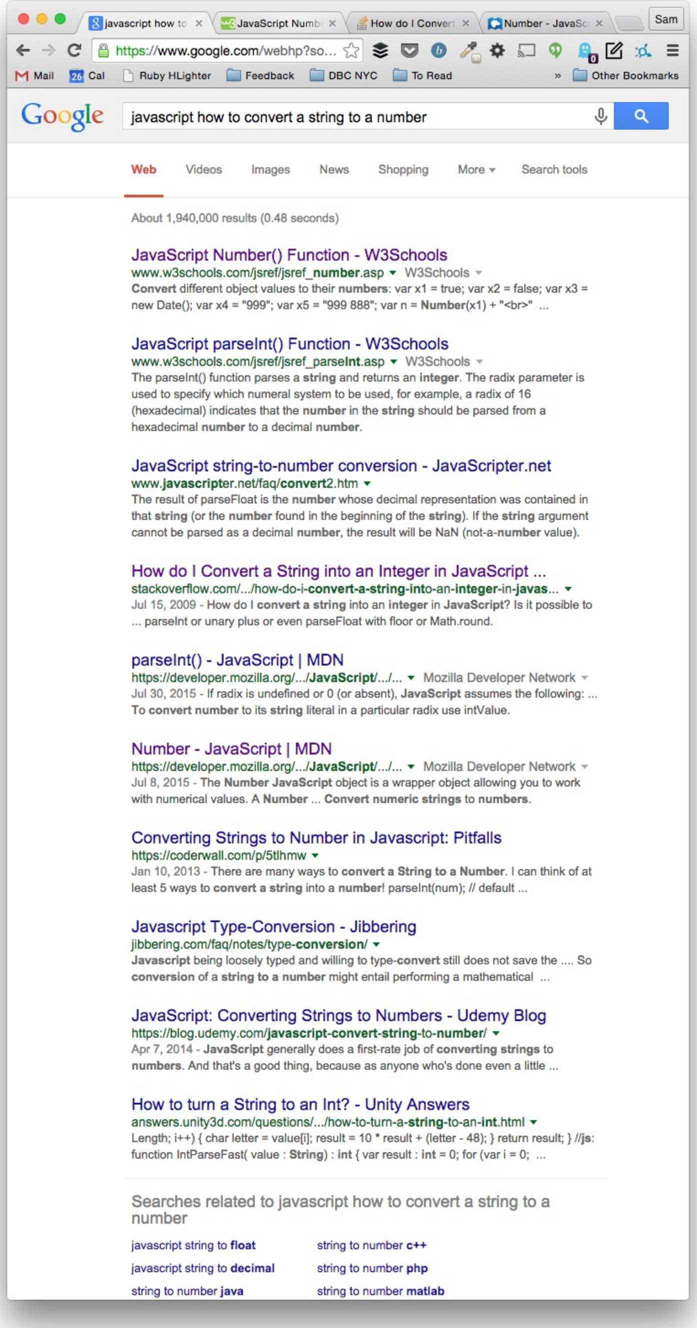
How to turn a String to an Int? - Unity Answers
answers.unity3d.com/questions/.../how-to-turn-a-string-to-an-int.html ▾
Length; i++) { char letter = value[i]; result = 10 * result + (letter - 48); } return result; } //js: function IntParseFast(value : String) : int { var result : int = 0; for (var i = 0; ...

Searches related to javascript how to convert a string to a number

javascript string to float
string to number c++
javascript string to decimal
string to number php
string to number java
string to number matlab

Research techniques

Cmd+Click (Mac) or Ctrl+Click a link to open it in a new tab. Open what you think are the three most promising results.



javascript how to convert a string to a number

About 1,940,000 results (0.48 seconds)

JavaScript Number() Function - W3Schools
www.w3schools.com/jsref/jsref_**number**.asp ▾ W3Schools ▾
Convert different object values to their **numbers**: var x1 = true; var x2 = false; var x3 = new Date(); var x4 = "999"; var x5 = "999 888"; var n = **Number**(x1) + "<br>" ...

JavaScript parseInt() Function - W3Schools
www.w3schools.com/jsref/jsref_**parseInt**.asp ▾ W3Schools ▾
The parseInt() function parses a **string** and returns an **integer**. The radix parameter is used to specify which numeral system to be used, for example, a radix of 16 (hexadecimal) indicates that the **number** in the **string** should be parsed from a hexadecimal **number** to a decimal **number**.

JavaScript string-to-number conversion - JavaScripter.net
www.**javascript**er.net/faq/**convert**2.htm ▾
The result of parseFloat is the **number** whose decimal representation was contained in that **string** (or the **number** found in the beginning of the **string**). If the **string** argument cannot be parsed as a decimal **number**, the result will be NaN (not-a-**number** value).

How do I Convert a String into an Integer in JavaScript ...
stackoverflow.com/.../how-do-i-**convert-a-string-into**-an-**integer-in-javas**... ▾
Jul 15, 2009 - How do I **convert a string** into an **integer** in **JavaScript**? Is it possible to ... parseInt or unary plus or even parseFloat with floor or Math.round.

parseInt() - JavaScript | MDN
https://developer.mozilla.org/.../**JavaScript**/.../... ▾ Mozilla Developer Network ▾
Jul 30, 2015 - If radix is undefined or 0 (or absent), **JavaScript** assumes the following: ... **To convert number** to its **string** literal in a particular radix use intValue.

Number - JavaScript | MDN
https://developer.mozilla.org/.../**JavaScript**/.../... ▾ Mozilla Developer Network ▾
Jul 8, 2015 - The **Number JavaScript** object is a wrapper object allowing you to work with numerical values. A **Number** ... **Convert numeric strings** to **numbers**.

Converting Strings to Number in Javascript: Pitfalls
https://coderwall.com/p/5tlhmw ▾
Jan 10, 2013 - There are many ways to **convert a String to a Number**. I can think of at least 5 ways to **convert a string** into a **number**! parseInt(num); // default ...

Javascript Type-Conversion - Jibbering
jibbering.com/faq/notes/type-**conversion**/ ▾
Javascript being loosely typed and willing to type-**convert** still does not save the .... So **conversion** of a **string to a number** might entail performing a mathematical ...

JavaScript: Converting Strings to Numbers - Udemy Blog
https://blog.udemy.com/**javascript-convert-string**-to-**number**/ ▾
Apr 7, 2014 - **JavaScript** generally does a first-rate job of **converting strings** to **numbers**. And that's a good thing, because as anyone who's done even a little ...

How to turn a String to an Int? - Unity Answers
answers.unity3d.com/questions/.../how-to-turn-a-**string**-to-an-**int**.html ▾
Length; i++) { char letter = value[i]; result = 10 * result + (letter - 48); } return result; } //**js**: function IntParseFast(value : **String**) : **int** { var result : **int** = 0; for (var i = 0; ...

Searches related to javascript how to convert a string to a number

javascript string to **float**	string to number **c++**
javascript string to **decimal**	string to number **php**
string to number **java**	string to number **matlab**

Research techniques

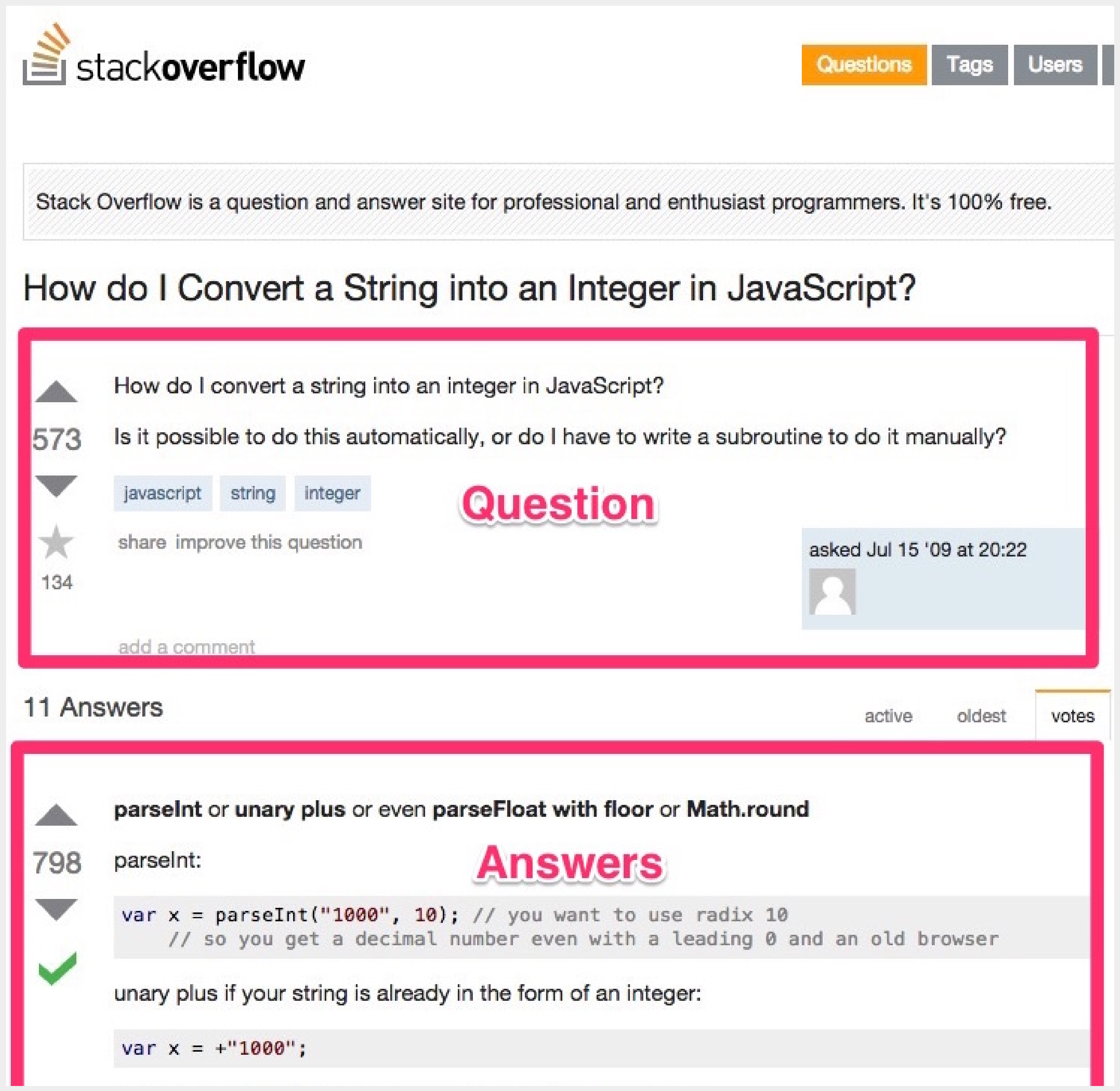
The following resources are particularly helpful:

- Stack Overflow
- MDN (Mozilla Developer Network)
- W3 Schools
- Guides and blog posts

Research techniques

A quick note on Stack Overflow...

Always remember that the section right at the top of the page is the question, not the answer!



Stack Overflow is a question and answer site for professional and enthusiast programmers. It's 100% free.

How do I Convert a String into an Integer in JavaScript?

573

How do I convert a string into an integer in JavaScript?

Is it possible to do this automatically, or do I have to write a subroutine to do it manually?

javascript string integer

134

share improve this question

asked Jul 15 '09 at 20:22

add a comment

Question

11 Answers

active oldest votes

798

parseInt or **unary plus** or even **parseFloat with floor** or **Math.round**

parseInt:

```
var x = parseInt("1000", 10); // you want to use radix 10
    // so you get a decimal number even with a leading 0 and an old browser
```

unary plus if your string is already in the form of an integer:

```
var x = +"1000";
```

Answers

Coding Basics

(through a JavaScript lens)

A bunch of jargon

- "variable"
- "string"
- "integer"
- "return-value"
- "array"
- "object"
- "property"
- "function"

What does it all mean!?

Variables

A variable is a way of **naming things** so we have a reference to them later.

Variables

Think of variables as a **label** we can write on and apply to different things.

Variables

Let's come back to variables once we learn about a few different **data types** we can label.

Strings

A string is simply a **sequence of characters**. In fact, this sentence itself could be a string. This is what it would look like in JavaScript:

```
"In fact, this sentence itself could be a string."
```

Strings can vary greatly in length, from entire novels...

```
"Well, Prince, so Genoa and Lucca are now just family estates of the
```

...to single words...

```
"hello"
```

...to nothing at all.

```
""
```

(this is known as an **empty string**)

Strings

Strings can contain numbers and don't have to make sense at all:

```
"9m52bqu4239wl"
```

Strings

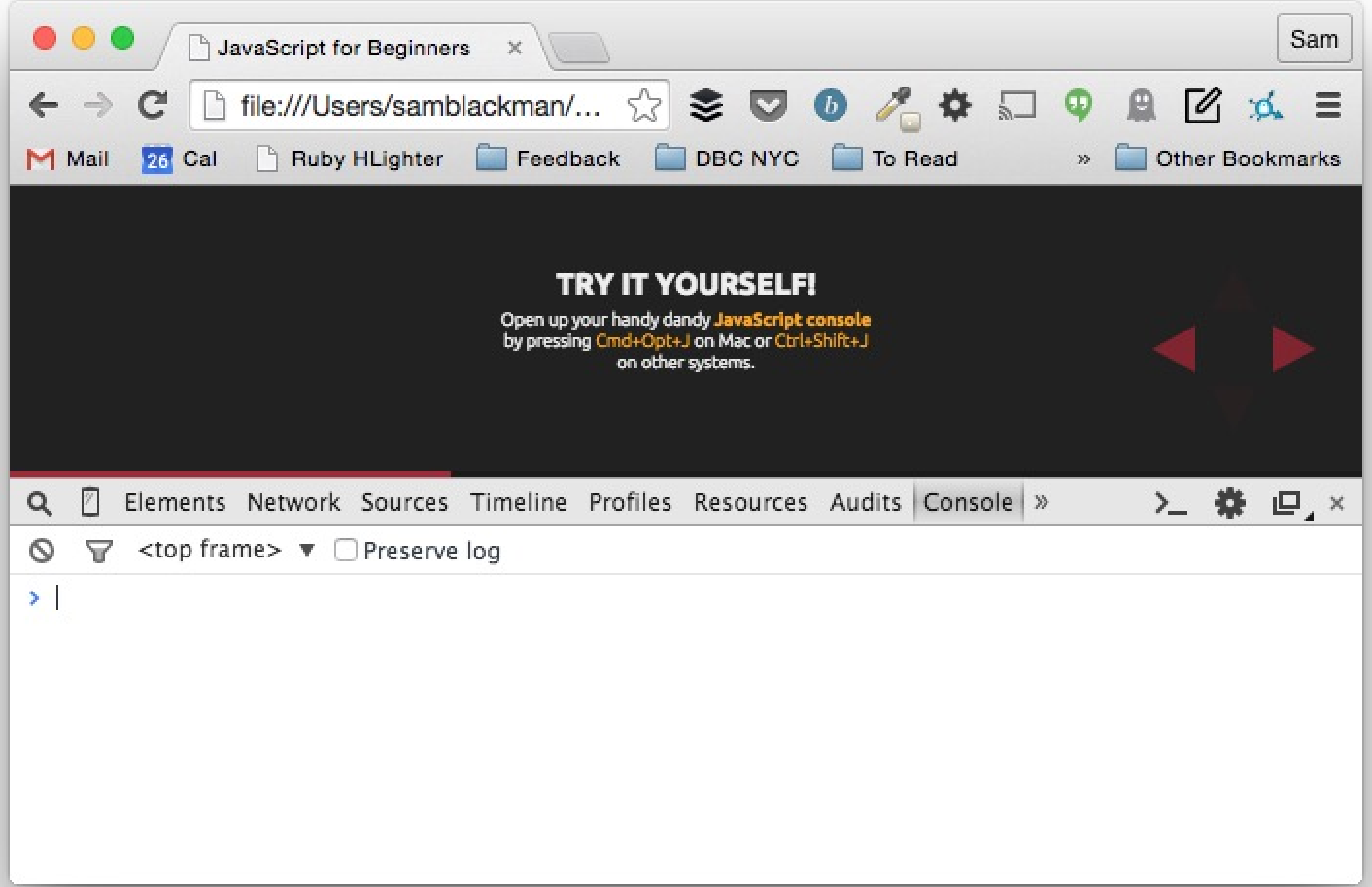
The **syntax** for creating strings in JavaScript is to wrap any number of characters in single or double quotes.

```
"This is a string."
```

```
'This is also a string.'
```

Try it yourself!

Open up your handy dandy **JavaScript console** by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J on other systems.



TRY IT YOURSELF!

Open up your handy dandy **JavaScript console** by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J on other systems.

Strings

Strings are one of the most common **data types** in every programming language. Get used to seeing, using and manipulating strings!

Numbers

Numbers are another common data type that you will see and use in the wild.

Numbers

The **syntax** for numbers in JavaScript is fairly intuitive.

```
42
3.14
100
10000
```

Numbers

You can perform simple arithmetic on numbers using:

Addition	+
Subtraction	-
Multiplication	*
Division	/
Modulus	%
Increment	++
Decrement	--

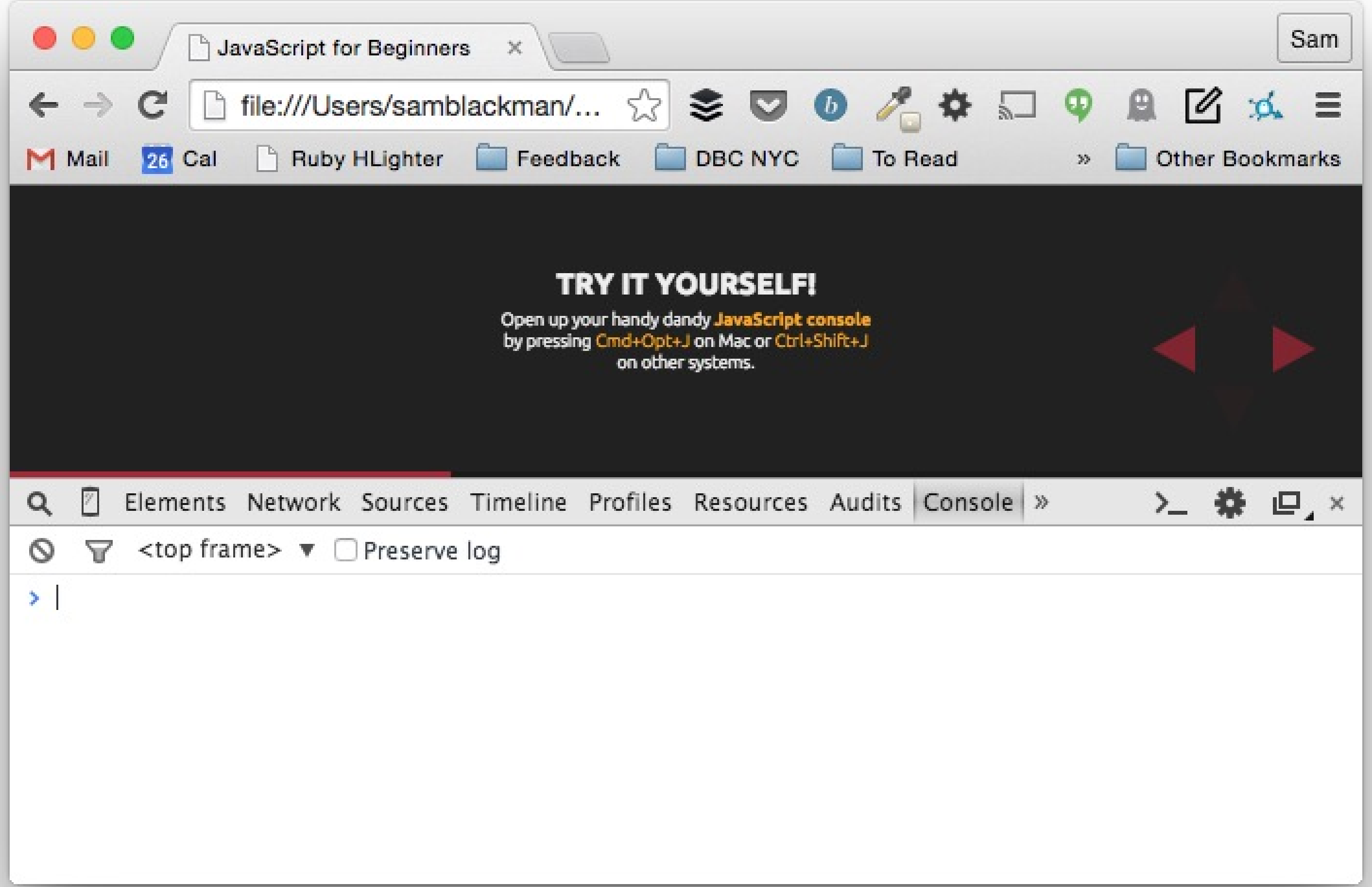
Numbers

You can also perform simple comparison operations on numbers using:

Equality	===
Inequality	!==
Greater than	>
Greater than or equal to	>=
Less than	<
Less than or equal to	<=

Try it yourself!

Open up your handy dandy **JavaScript console** by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J on other systems.



TRY IT YOURSELF!

Open up your handy dandy **JavaScript console** by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J on other systems.

Return values

Every time you press enter in the JavaScript console, you see the **return value** of expression you evaluated.

What data type did JavaScript return when you used a comparison operator?

Comparison operators will always return a

Boolean

(true or false)

Back to variables

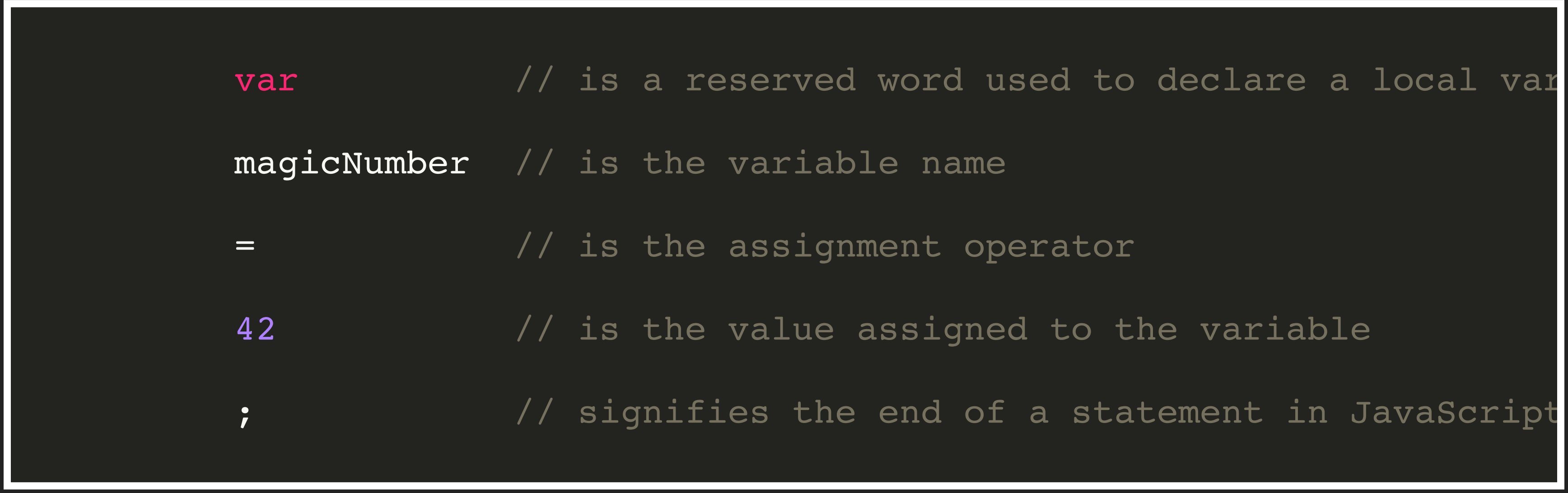
Now we know how to create a few data types, let's save them for later in some **variables**.

Variable syntax

The syntax for **declaring a variable** in JavaScript is like this:

```
var magicNumber = 42;
```

Breaking it down



```
var           // is a reserved word used to declare a local var

magicNumber   // is the variable name

=             // is the assignment operator

42            // is the value assigned to the variable

;             // signifies the end of a statement in JavaScript
```

A note on naming

Naming variables in **camelCase** is a JavaScript convention.

Storing things

You can directly assign values to **variables**...

```
var seenIt = "...like we've already seen";
```

...or store **return values** of expressions in **variables**

```
var num = 3;
var otherNum = 4;
var multiplier = 6;
var answerToTheUltimateQuestion = (num + otherNum) * multiplier;
```

Data structures

Let's look at two more **data structures**:

- Arrays
- Objects

Data structures

For the following data structures, we are going to learn how to create, or **instantiate**, them and then learn the following operations:

- Accessing values
- Updating values
- Inserting values
- Removing values

Arrays

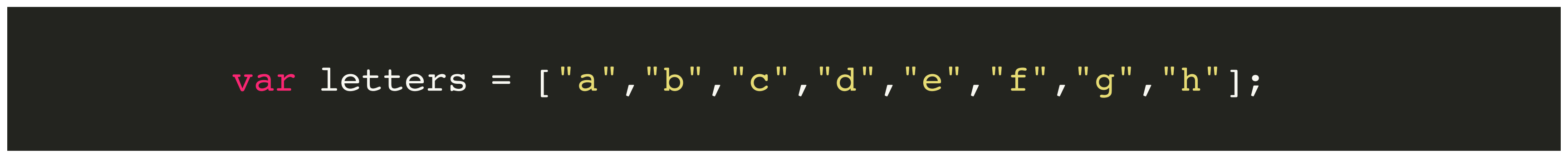
Just like a **variable** holds a single value, an **array** holds a collection of values.

Arrays

Think of an array as a bunch of buckets, each of which stores a value.

Arrays

If we were to put the first eight characters of the alphabet into these buckets, it would look like this in JavaScript:



```
var letters = ["a","b","c","d","e","f","g","h"];
```

An **array** of the characters "a" through "h" is now stored in the variable `letters`. This is an array of **strings**.

Accessing values

Every element in an array is stored in a specific position known as an **index**. Arrays are indexed starting at 0 and incrementing by 1 for each position.

0	1	2	3	4	5	6	7
"a"	"b"	"c"	"d"	"e"	"f"	"g"	"h"

Arrays

You refer to each **element** by its **index**.

Arrays

Accessing values

```
var letters = ["a","b","c","d","e","f","g","h"];
letters[0]// -> "a"
letters[7]// -> "h"
```

Arrays

Updating values

```
var letters = ["a","b","c","d","e","f","g","h"];
letters[0] = "apples"  // -> "apples"
letters[1] = "oranges" // -> "oranges"
letters      // -> ["apples","oranges","c","d","e","f","g","h"]
```

Arrays

Inserting values

```
var letters = ["a","b","c","d"];
letters.push("elephant") // -> "elephant"
letters // -> ["a","b","c","d","elephant"]
```

The **push** function adds an element to the end of an array.

Arrays

Inserting values

```
var letters = ["a","b","c","d", "elephant"];
letters.unshift("zebra") // -> "zebra"
letters // -> ["zebra","a","b","c","d", "elephant"]
```

The **unshift** function adds an element to the start of an array.

Arrays

Removing values

```
var letters = ["a","b","c","d"];
var char = letters.pop() // -> "d"
letters // -> ["a","b","c"]
```

The **pop** function removes an element from the end of an array.

Arrays

Removing values

```
var letters = ["a","b","c","d"];
var char = letters.shift() // -> "a"
letters // -> ["b","c", "d"]
```

The **shift** function removes an element from the start of an array.

Functions

All data types in JavaScript come with some built in behavior called **functions**. We've already seen a few array functions with push, pop, shift and unshift.

Function syntax

```
var letters = ["a","b","c"];
letters.push("d");
// --------------------------

letters // the array that we're calling our function on
.       // the dot to signify we're about to call a function
push    // the name of the function
("d")   // the actual calling of the function with a string argument
```

We **call a function** with a dot, followed by the function name, followed by a set of parentheses.

Sometimes we put other data inside the parens, known as **arguments**; other times functions don't require arguments.

Properties

Things in JavaScript can also have **properties**. You access and modify properties similarly to functions, except you do not include the parentheses.

Properties

```
var fruits = ["apple","banana","carrot"];
fruits.length // -> 3
fruits[2].length // -> 6
fruits.push("d");
fruits.length // -> 4
```

length is a good example of a **property** on both arrays and strings. Note how properties don't need parentheses after them.

Functions vs Properties

It can be very confusing to know whether something is a **function** or an **property** when you start out with JavaScript. It's something you'll get used to.

Remember you always have the Web to look things up!

Objects

Objects in JavaScript are similar to arrays in that they contain a collection of values.

However, unlike array values which are **accessed and ordered by index**, the values in objects are known as properties are accessed via **property names**

Objects

In our metaphor we give each bucket a name. This is the **property name**. The item inside the bucket is its **value**. So, a JavaScript object is a collection of **named properties**.

BON VOYAGE
BY RAIL AND SEA TO FRANCE AND THE CONTINENT
name
age
sex
height
eyes
hair
phone
email

Objects

Let's **instantiate** an object in JavaScript:

```
var person = {name: "Sam", age: 28, sex: "male"};
```

This **object** contains three **properties** and the object is stored in the variable `person`.

Objects

Accessing values

```
var person = {name: "Sam", age: 28, sex: "male"};
person.name      // -> "Sam"
person["name"]   // -> "Sam"
```

Objects

Updating values

```
var person = {name: "Sam", age: 28, sex: "male"};
person.age           // -> 28
person.age = 29      // -> 29

// person now contains
{name: "Sam", age: 29, sex: "male"}

person.age           // -> 29
```

Objects

Inserting values

```
var person = {name: "Sam", age: 28, sex: "male"};
person.gender = "cis male"  // -> "cis male"

// person now contains
{name: "Sam", age: 28, sex: "male", gender: "cis male"}
```

Objects

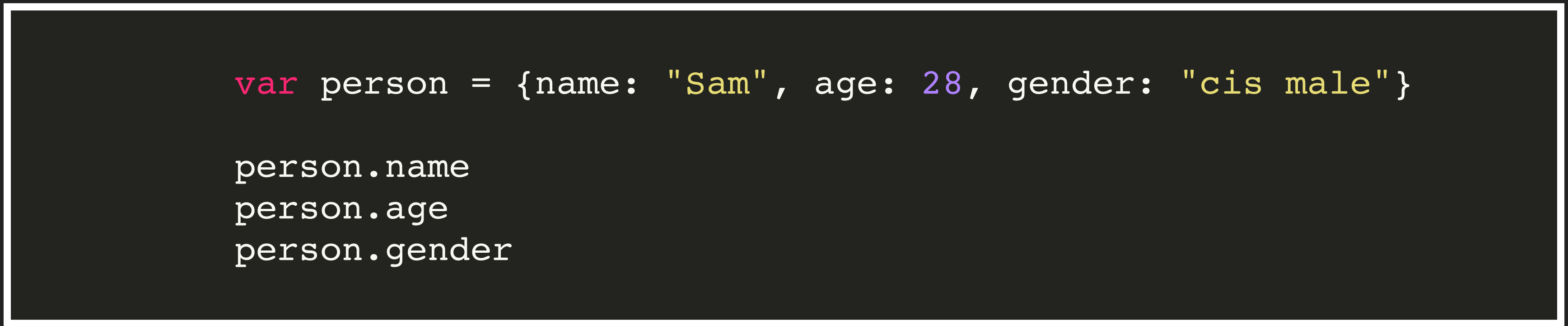
Removing values

```
var person = {name: "Sam", age: 28, sex: "male", gender: "cis male"}
delete person.sex // -> true

// person now contains
{name: "Sam", age: 28, gender: "cis male"}

person.sex        // -> undefined
```

The **delete** operator removes a **key value pair** from an object.

Taxonomy

We've been using **dot notation** when working with objects.



```
var person = {name: "Sam", age: 28, gender: "cis male"}

person.name
person.age
person.gender
```

What type of thing does that suggest `name`, `age`, and `gender` are?

properties

Taxonomy

So what we generically call **keys** are specifically called **properties** in JavaScript objects.

Also, these things we call **objects** in JavaScript can be called dictionaries, hashes, hash tables, or maps in other languages.

omg let's build.

bit.ly/dbc-js-intro

With your neighbor, model the following using the data types and structures you just learned:

- A list of three different people
- The people should have names, ages and a list of their three favorite foods.

What is the best way to do this?

- Arrays?
- Objects?
- Arrays of Objects?

Enter your data structure into the console and work through any errors.

Loops, Conditionals, & Functions

Loops

Doing certain things over and over and over and over is a very common thing when coding

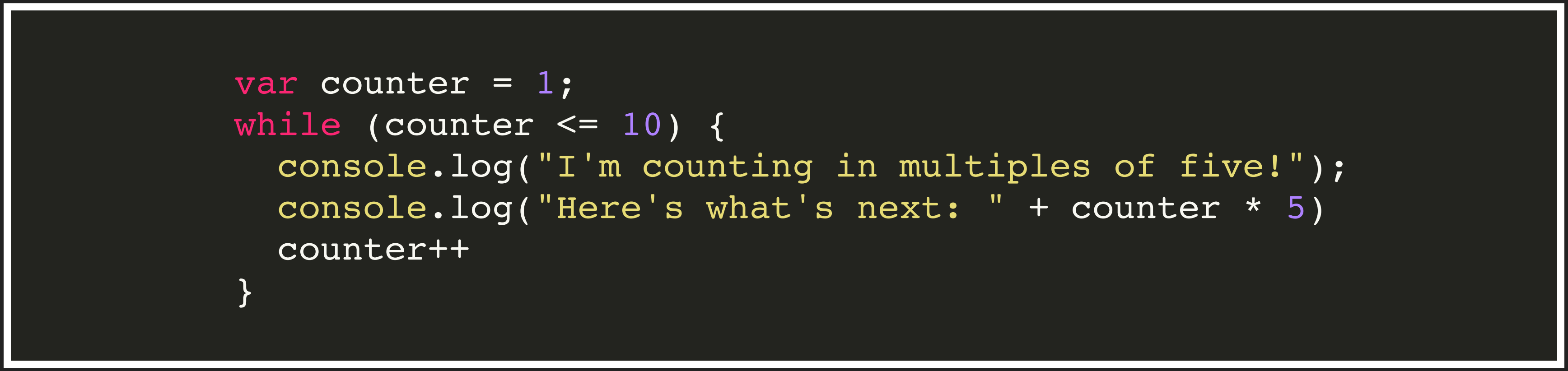
Loops

We will look at the syntax for two JavaScript **loops**:

- for
- while

Loops

This is the syntax for a **while loop**.



```
var counter = 1;
while (counter <= 10) {
  console.log("I'm counting in multiples of five!");
  console.log("Here's what's next: " + counter * 5)
  counter++
}
```

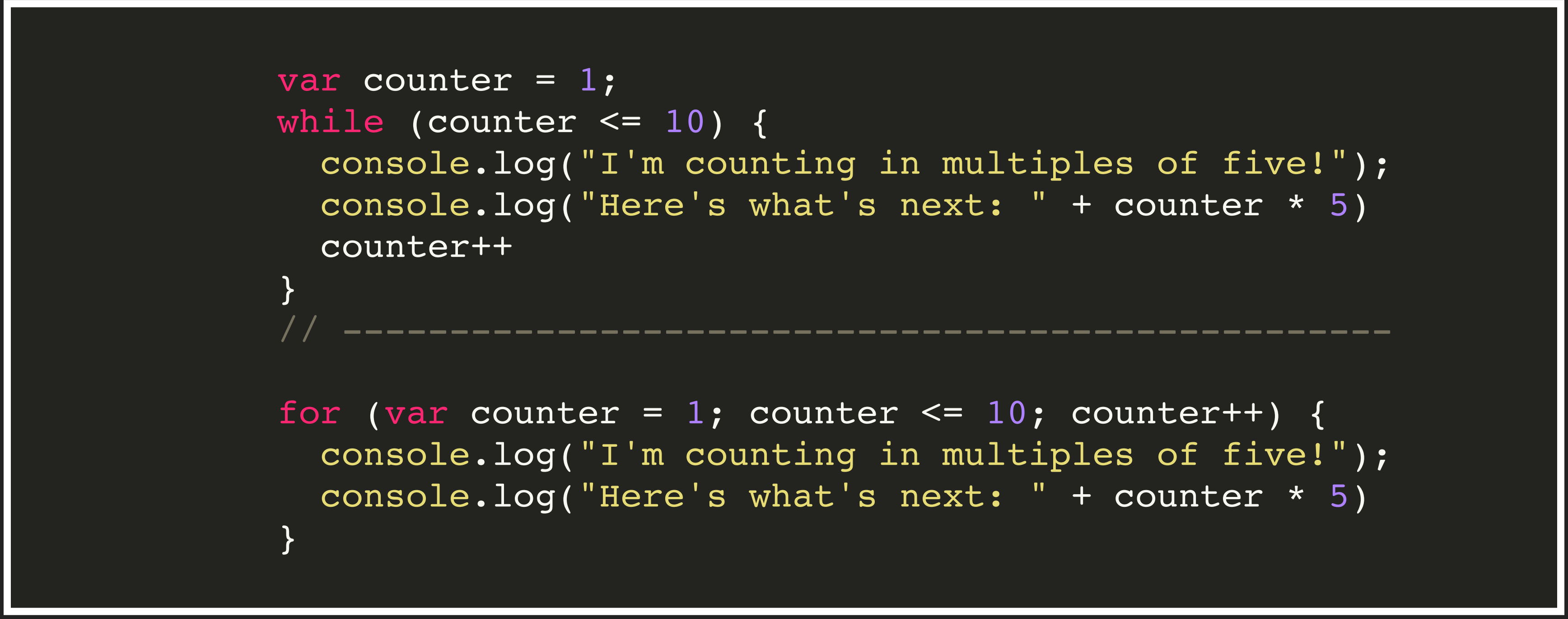
Loops

Here's the same operation using a **for loop**.

```
for (var counter = 1; counter <= 10; counter++) {
  console.log("I'm counting in multiples of five!");
  console.log("Here's what's next: " + counter * 5)
}
```

Loops

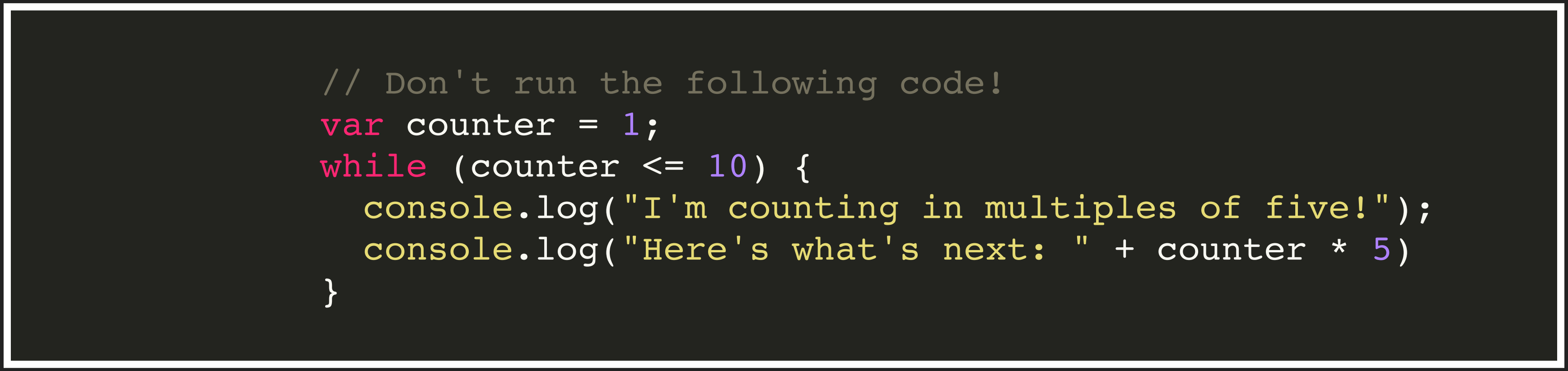
Compare the two:



```
var counter = 1;
while (counter <= 10) {
  console.log("I'm counting in multiples of five!");
  console.log("Here's what's next: " + counter * 5)
  counter++
}
// -------------------------------------------------

for (var counter = 1; counter <= 10; counter++) {
  console.log("I'm counting in multiples of five!");
  console.log("Here's what's next: " + counter * 5)
}
```

Warning!

Be careful not to code a loop that will never end. It will cause your browser to freeze and crash!



```
// Don't run the following code!
var counter = 1;
while (counter <= 10) {
  console.log("I'm counting in multiples of five!");
  console.log("Here's what's next: " + counter * 5)
}
```

This is called an **infinite loop**. Can you see why it never ends?

Loops

Here's a fun game with a **while loop**.

```
var secretPhrase = "bananas"
var userInput
while (userInput !== secretPhrase) {
  userInput = prompt("Haha! You will continue to get this annoying\
  pop up until you guess the secret phrase!")
}
alert("Drat! You guessed it!")
```

Loops

The most common use of loops is to **loop over collections**. We loop over collections when we want to do something with every element in a collection.

Loops

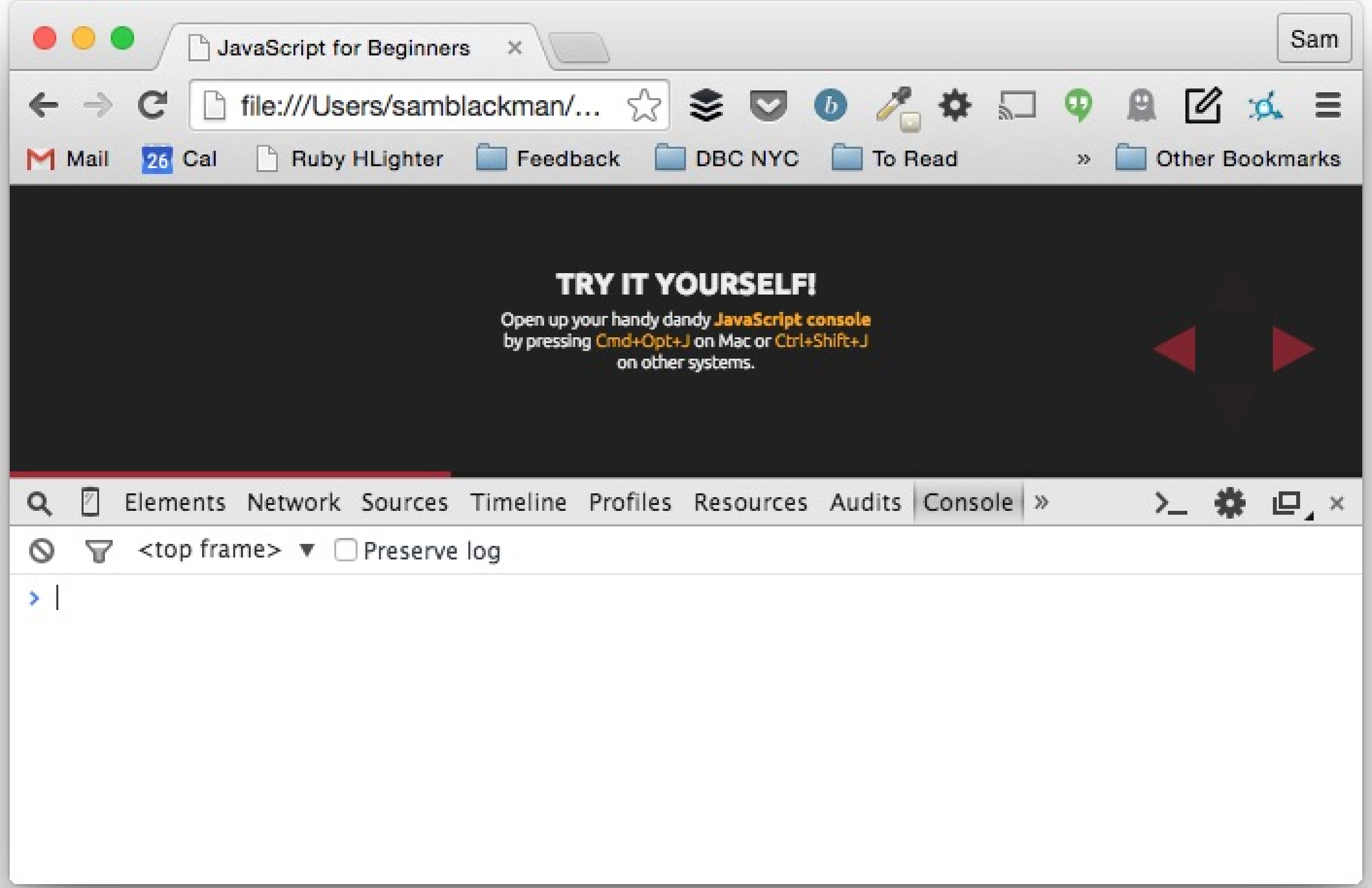
Here's a **for loop** looping over and printing out every element in our letters array:

```
var letters = ["a","b","c","d","e","f","g","h"];
for (var i = 0; i < letters.length; i++) {
  console.log(letters[i]);
}
```

There's a lot going on here. So let's talk through it.

Try it yourself!

Open up your handy dandy **JavaScript console** by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J on other systems.



TRY IT YOURSELF!

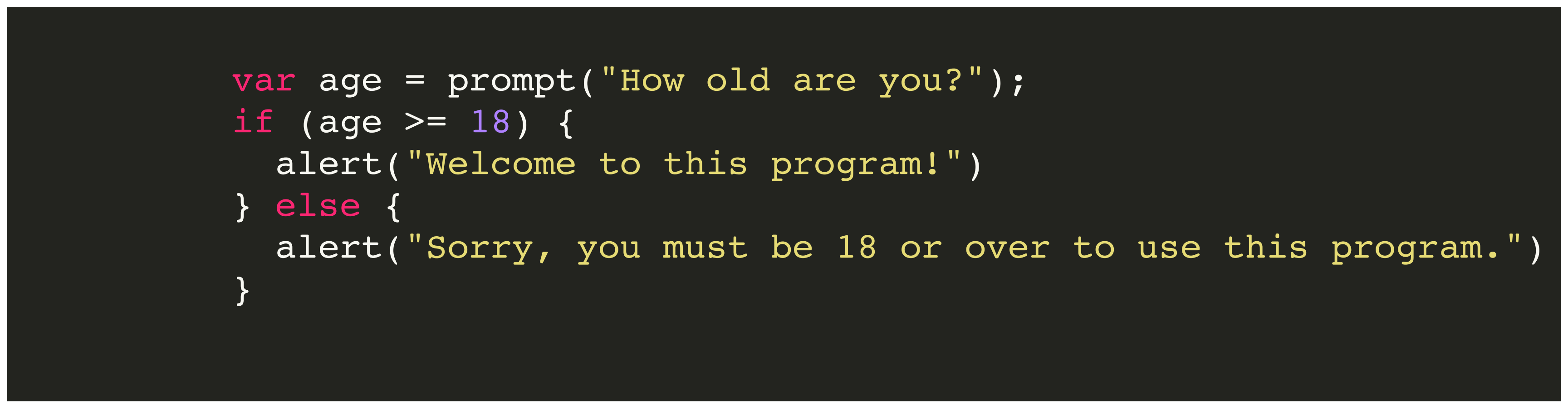
Open up your handy dandy JavaScript console by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J on other systems.

Conditionals

Sometimes you want your code to do different things depending on different inputs. This is called **control flow**.

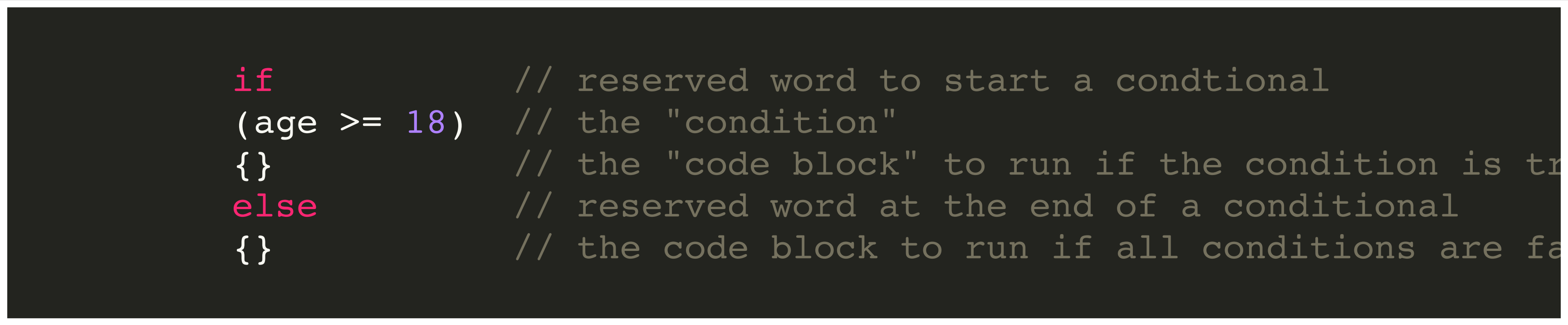
Conditionals

Maybe you need to check someone's age before you let them use your program:



```
var age = prompt("How old are you?");
if (age >= 18) {
  alert("Welcome to this program!")
} else {
  alert("Sorry, you must be 18 or over to use this program.")
}
```

Breaking it down



```
if             // reserved word to start a condtional
(age >= 18)    // the "condition"
{}             // the "code block" to run if the condition is tr
else           // reserved word at the end of a conditional
{}             // the code block to run if all conditions are fa
```

What is a function?

A **function** is a selection of code that you can save and run later, potentially multiple times.

Function syntax

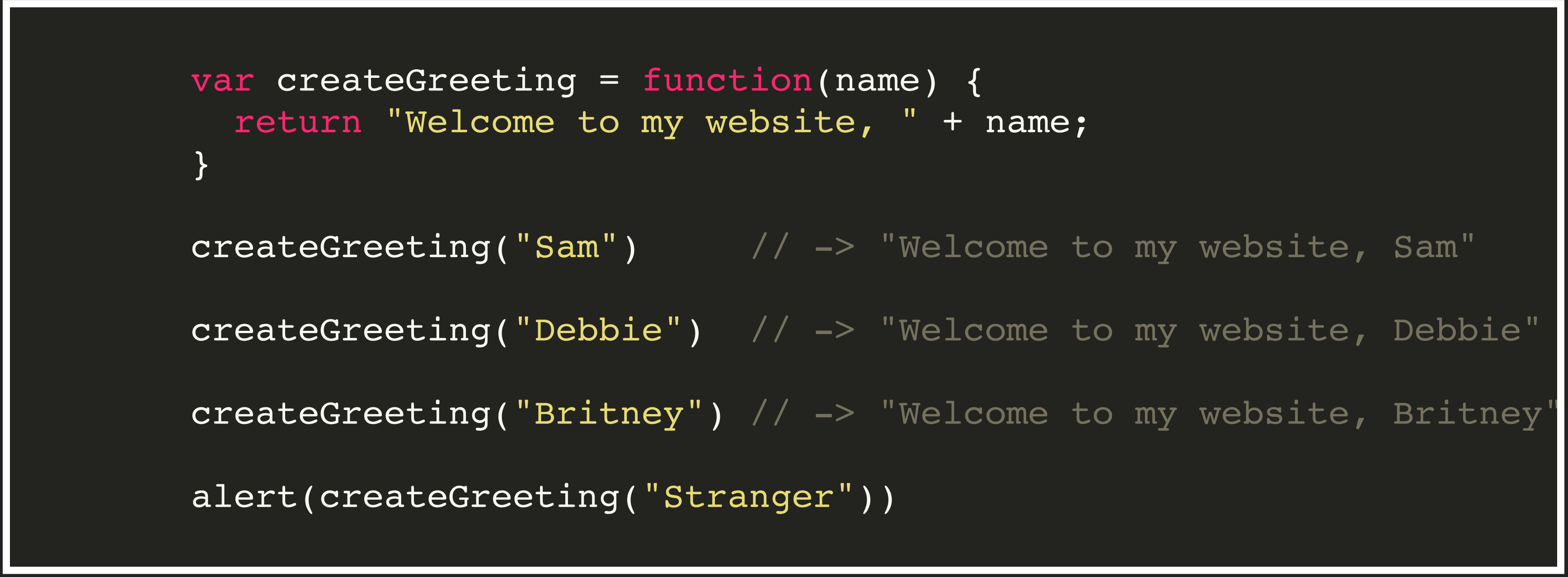
We **define a function** like this:

```
var createGreeting = function(name) {
  return "Welcome to my website, " + name;
}
```

The code is now stored in the variable `createGreeting`.
Now we can call this the **createGreeting** function.

Calling a function

We **call a function** by typing its name, followed by parentheses.



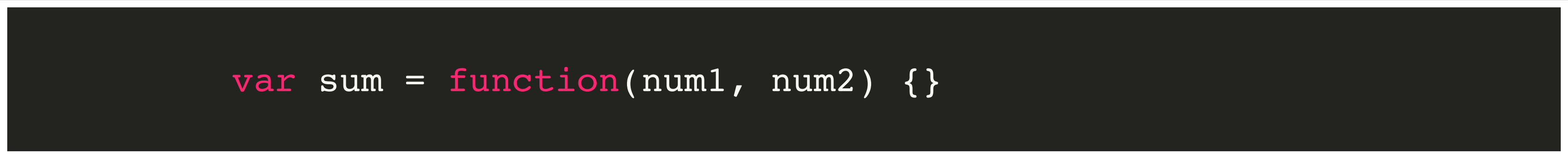
```
var createGreeting = function(name) {
  return "Welcome to my website, " + name;
}

createGreeting("Sam")      // -> "Welcome to my website, Sam"

createGreeting("Debbie")   // -> "Welcome to my website, Debbie"

createGreeting("Britney") // -> "Welcome to my website, Britney"

alert(createGreeting("Stranger"))
```

Input and Output

Functions allow us to **input** some data and **output** other data.

Arguments

The input that we give to functions are called **arguments**.



```
var sum = function(num1, num2) {}
```

`num1` and `num2` are the **parameters** in this function that show us that we can pass it two arguments.

Return values

The output from a function is its **return value**.

```
var sum = function(num1, num2) {
  var result = num1 + num2;
  return result; // return value!
}
```

Usually we manipulate our input in some useful way and then **return** that data.

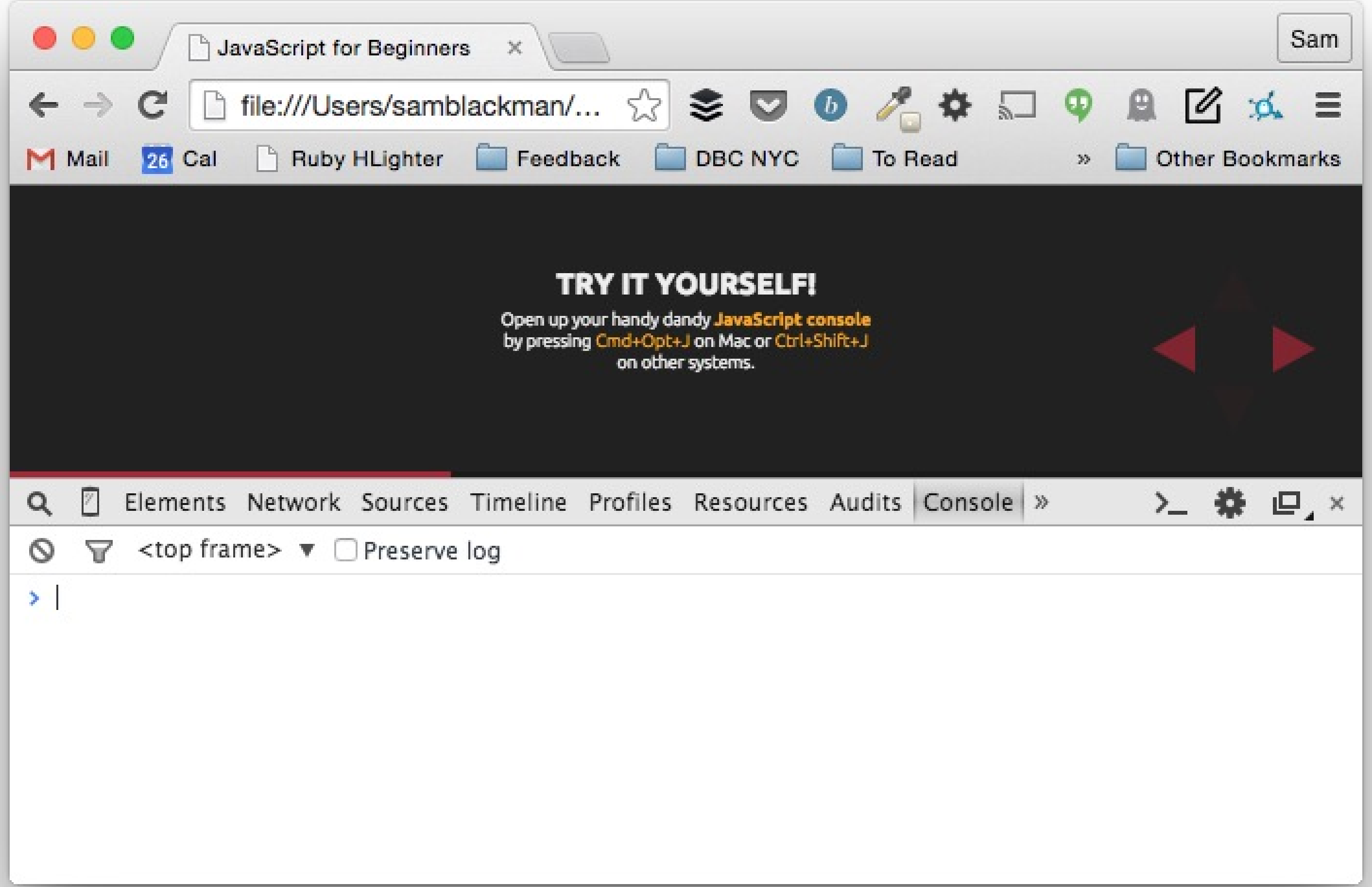
Functions

Functions **don't actually need** parameters or a return statement. Here's an example:

```
var woohoo = function() {
  alert("WOOHOO!");
}
```

Try it yourself!

Open up your handy dandy **JavaScript console** by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J on other systems.



TRY IT YOURSELF!

Open up your handy dandy **JavaScript console** by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J on other systems.

Intro to jQuery

jQuery

A JavaScript library that makes it easier for us to interact with webpages.

Some new files!

bit.ly/dbc-jquery-intro

Loading jQuery

In the index.html file, you'll see a **<script> element** linking to a copy of the jQuery in the same folder. This is one way to include jQuery on your page.

You'll also see a <script> element that is commented out. This is another way of including jQuery: by linking to an external source.

Selecting elements

We are going to be playing with the **HTML elements** on our page. To do that, we first need to **select** those elements using jQuery.

Selecting elements

The syntax for **selecting elements** with jQuery is:

```
$("div")
```

This will select all of the div elements on the page.

Selecting elements

You can use any **CSS selection** syntax with jQuery:

```
// to select all paragraph elements:
$("p")

// to select all elements with the class 'shadow':
$(".shadow")

// to select the element with the id 'main-container':
$("#main-container")
```

Just remember to wrap your **selector** in quote marks to make it a **string**.

jQuery functions

Once we have selected elements, we can **call jQuery functions** on those elements:

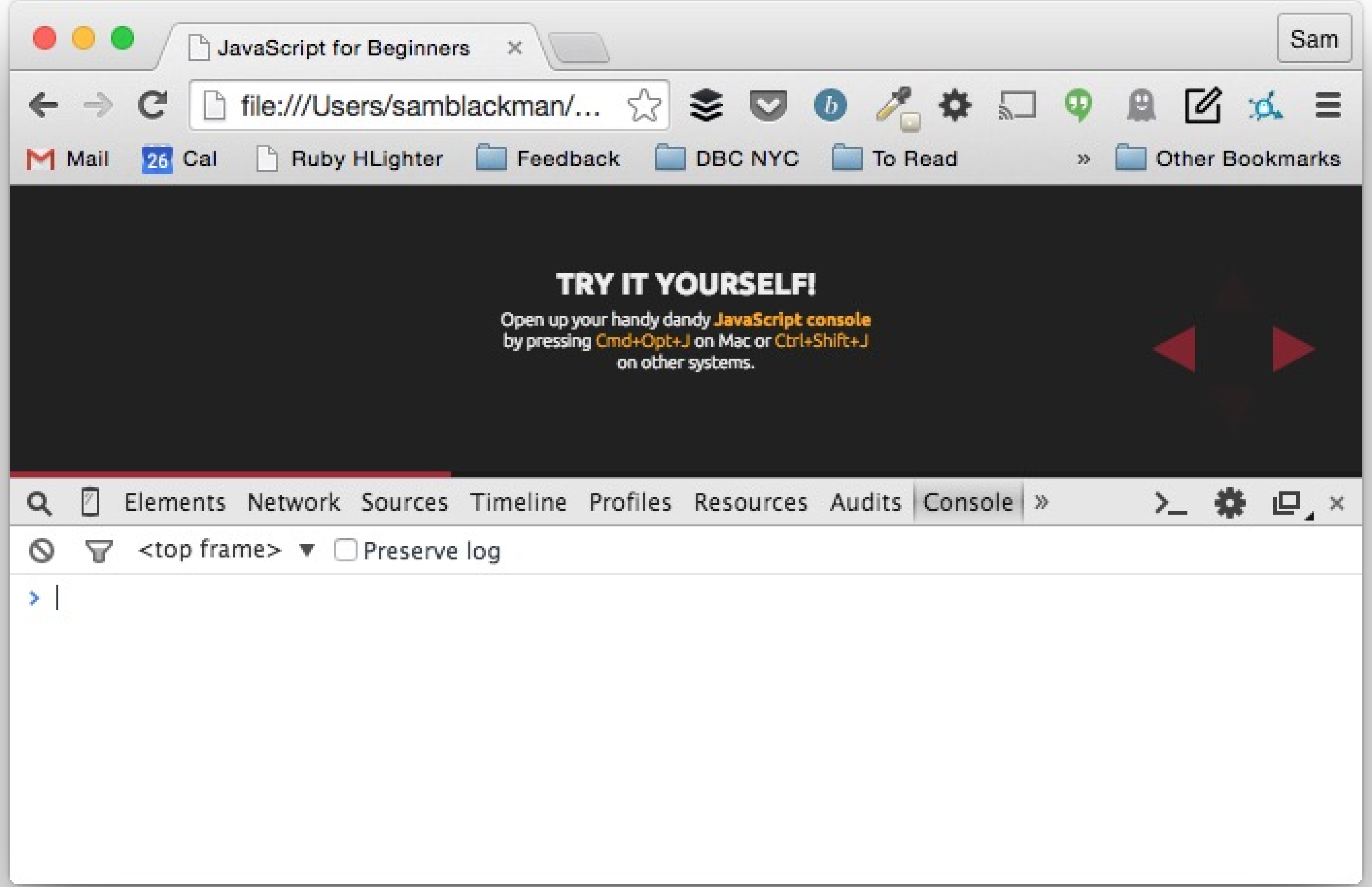
```
$("p").css("background-color", "aqua");
```

This selects all paragraphs and then turns their background colors **aqua!**

Try it yourself!

Open up your handy dandy **JavaScript console** by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J on other systems.

Make sure you open it up on the **index.html** page that has jQuery loaded!



TRY IT YOURSELF!

Open up your handy dandy **JavaScript console**
by pressing Cmd+Opt+J on Mac or Ctrl+Shift+J
on other systems.

Listening for events

Interacting with a website through **clicks** and **keystrokes** really brings JavaScript (and websites) to life!

Let's learn how to tell certain elements on our page to listen for a click event.

Listening for events

First we need to **select an element**. Let's select our button:

```
$("#load-tweets-button")
```

Listening for events

Now we need to **bind a click event** to our button:

```
$("#load-tweets-button").click(function() {
  alert("Button clicked!");
});
```

What this says is: 'whenever this thing I have selected is clicked, run the code in this function'.

Listening for events

One very common thing to do when a button is clicked is to **update something else on the page**.

```
$("#load-tweets-button").click(function() {
  // your code here that updates something else on the page, maybe:
  $("p").css("background-color", "aqua");
});
```

Now whenever you click the button, all the paragraphs will turn **aqua!**

Updating the page

Here's one more super helpful **jQuery function** called `append`:

```
$("#main-container").append("<p>A friendly paragraph!</p>")
```

The append function will append whatever **HTML string** you pass it as an argument to the element(s) you originally **selected**.

Updating the page

Remember, you can add things to strings like this:

```
var dataFromSomewhere = "A friendly paragraph!"
$("#main-container").append("<p>" + dataFromSomewhere + "</p>")
```

This is called **string concatenation**.

Putting it all together

You now have all the tools to dynamically create a webpage! It might strain your brain, but it's time to put **everything we learned into practice**.

omg let's build.

bit.ly/dbc-jquery-intro

Task: when you click the button all of the tweets will appear in new paragraphs on the page.

You will need to use:

- Loops (for or while)
- Array functions and properties (.length)
- Object functions and properties
- jQuery selectors - $('div')
- jQuery event listeners - .click()
- the jQuery append function

Resources to learn more

- Codecademy
- Code School
- Treehouse
- Stack Overflow
- jQuery Documentation
- MDN (Mozilla Developer Network)
- W3 Schools
- Guides and blog posts
- Google
- Playing around!

Thanks for joining us!

Please give us feedback:

bit.ly/intro-js-feedback