**A JavaScript Crash Course - Introduction and Hello World -** [James Bryce](https://jybryce04.medium.com/?source=post_page-----33fd4b0ecfeb--------------------------------) Jan 5, 2022

Part 1: *The first part of my series on JavaScript Fundamentals*

*This course assumes you have installed a text editor, preferably Visual Studio Code, Google Chrome, and possess working knowledge of basic HTML/CSS. It would help to install Node.js ahead of time, but it isn’t required.*

**Introduction and Course Structure**

Welcome to the first of many articles where I teach you some basics of web development! This is a course on JavaScript — the scripting language that powers the World Wide Web. Traditionally, one would start with HTML and CSS, but exposure to JavaScript early on in your learning will better prepare you for more advanced topics.

Make no mistake, though, *you will need to know HTML/CSS* to understand many things, and those topics are foundational to all things front-end. However, if you are coming from a systems perspective or backend work, such as working with C/C++, Java, or other traditionally desktop stand-alone applications programming languages, know that you’re not alone.

The first programming language I learned was C++ when I was still in high school, and I didn’t think seriously about JavaScript until it became apparent how prevalent it was and will continue to be.

**About Me**

My name is James, and I’m from Austin, Texas. I have been studying computer programming on my own for a long time and at Austin Community College for a few years. I know HTML/CSS and JavaScript, Python, C++, Java, relational databases, and related topics. I don’t pretend to know everything, but I consider myself a lifelong learner willing to grow and make mistakes.

That’s the attitude I think developers should adopt — a passion for learning new things and a willingness to accept they aren’t perfect. It makes things easier if you don’t expect to be the smartest or write the cleanest and most elegant code, especially when you’re a beginner in this vast field.

**Course Outline**

We will be covering each of the topics in the list below. I’ll add links to each article as they are published. I strongly recommend following the order of the course as each topic builds on the last — except for the optional history and high-level explanation.

* Introduction, Hello World, and Variables (This article)
* A Brief History of JavaScript (optional, but it provides context)
* High-Level Explanation of How JavaScript Works (stuff you may find interesting)
* Data Types, var, let, and const
* More About Arrays, Strings, and Objects
* Logical/Arithmetic Operators; Conditionals and Loops
* Functions, Scope, and variable hoisting; reference vs. value
* Exception Handling
* IIFE, Closures, and Callbacks
* ES6 Classes and OOP

Along the way, we’ll learn how to implement some classic data structures and algorithms with JavaScript, some features introduced in ES6+, and look ahead toward more advanced concepts that I’ll be writing about in detail in the future. I’ll also be including a few challenges of varying difficulty at the end of each article for you to test your skills! Additionally, I will provide links as needed to more resources such as other articles, YouTube videos, free courses, and documentation to enhance your understanding.

Note that this is not a comprehensive course but one of many potential starting points for the aspiring web developer.

With that out of the way, let’s write our first lines of JavaScript!

**Hello, World!**

Traditionally, the first lines of code you write in any language accomplish one thing: outputting the text, “Hello, World!” to the screen. This tradition dates back to the famous book [*The C Programming Language*](https://www.amazon.com/Programming-Language-2nd-Brian-Kernighan/dp/0131103628/ref=sr_1_1?crid=1FLU0ZD27M1S0&keywords=the+c+programming+language+by+kernighan+and+ritchie&qid=1641325145&sprefix=The+C+P%2Caps%2C114&sr=8-1)*.*Since then, virtually everyone has written a HelloWorld program to start their code writing voyage.

Open Visual Studio Code (VSC), make a new file, save it as **hello.js**, and type console.log(‘Hello, World!’); in it. You can then save this file and run it in one of two ways:

**Run JavaScript in Chrome**

Make an HTML file and link to **hello.js** as an src attribute in a script tag. For example,

<!-- HTML document above --><script src="hello.js"></script><!-- The rest of the HTML document -->

Open the HTML file in Chrome, open the developer tools by pressing ⌘+⌥+I on macOS or Ctrl+Shift+I on Windows. Look at the *console*. It should show the text “Hello, World!”. This is what you should see:

A screenshot of a computer

Description automatically generated with medium confidence

My Chrome Developer Tools Open showing the result of the HelloWorld script in the console

**Run JavaScript with Node.js**

If you installed [Node.js](https://nodejs.org/en/) ahead of time, you could run **hello.js**much more quickly. For simplicity, you could use VSC’s Integrated Terminal (Ctrl+Shift+`) then type at the prompt node hello.js and press the return/enter key.

Graphical user interface, text, application

Description automatically generated

Running the JavaScript Hello World with Node.js. I am very creative with my example directory names…

Above, you’ll see that it did not take much to run the script in the terminal (out of habit, I called my file **app.js**, but you can name it **hello.js** or **hello\_world.js** or anything you want). That’s why I suggested you install Node.js. Since we’re going to be working a lot with console.log as we learn the basics, it is far easier to use Node.js for running these scripts. Note that **you are not ready to learn more about Node.js until you learn the fundamentals of JavaScript**. Nevertheless, I will take advantage of this feature of Node throughout this tutorial series for many examples.

**Declaring and Initializing Variables**

Now that we’ve written a line of JavaScript let’s change things up slightly. Instead of hard-coding the *string literal*“Hello, World!”, let’s store it in a variable called hello then pass that variable to console.log so we can display it in the terminal. Here’s how we do that:

var hello = 'Hello, World!';

Here’s what I have in my **app.js** file:

Graphical user interface, text, application

Description automatically generated

A screenshot of my app.js file showing you precisely what to do

On line 1, that’s a comment. The JavaScript engine ignores comments. Comments help document your code, explain things to your future self or other developers, and temporarily remove code when trying new things, as in this case. Don’t fret too much about it for now, but try to write comments according to the rules set by your organization or at least with consistency.

Notice also the keyword var. Later, we’ll learn about let and const, introduced in ES6, which in my opinion should almost always be used instead of var. For now, though, using var is permissible as we are working in the *global scope.*Don’t worry about what that means at present; in my data types and functions articles, I will explain these terms in detail.

Just to prove that this still works, here is the result of running the new **app.js**:

Graphical user interface, text, application

Description automatically generated

Running app.js with Node.js in my terminal

That’s all I have for you right now. Next, you can either go to the optional history or high-level explanation articles\* or skip straight to Data Types, var, let, and const\*.

Before you go, let’s review what you’ve learned:

* How to output text to the JavaScript console with console.log
* How to initialize a variable with the var keyword
* How to write a single-line comment

Here is a simple set of instructions that you can implement in JavaScript for practice if you feel so inclined:

1. Store the value of “Hello, hello! I am a web developer.” in a variable called greeting.
2. Output greeting to the JavaScript console.

Bonus exercise:

Write a script that stores a greeting and your name in two separate variables and outputs both to the console in the form **greeting, name**. Hint: The + symbol can combine strings (this is called *concatenation,* and you’ll learn an even better way to do this later).