JavaScript for App Development

By Mark Lassoff, Founder Framework Television

Section One

Welcome to the **JavaScript for App Development** program. I'm really glad that you're here.

This is a comprehensive program to prepare you for creating applications using the increasingly popular JavaScript Programming language. While program is suitable for beginners, it will move quickly from the basics in to more applicable topics that are used every day by professional JavaScript developers.

For each section of the course you'll be provided with a guide for the content and activities to be completed. When you've completed everything you'll move on to the next section of the course.

For the 2019 edition of this program, we have updated our code and instruction to be compliant with the ES6 standard. However, for the first few lessons we'll be sticking to traditional JavaScript as we have found that easier for most new developers to learn. You will find that, for the most part, ES6 is a superset of the traditional JavaScript language, meaning that everything you learn about "traditional" JavaScript in the first few sections will continue to be applicable even after we engage fully with ES6.

JavaScript Specialist Certification

Completing this course will make you eligible to earn the JavaScript Specialist Designation from Framework Television. This certification is designed to represent a fundamental understanding of the JavaScript programming language and it's applicability to web development and mobile development. There are three steps to certification:

1. Complete all of the Sections in the JavaScript for App Development course. You'll complete a section by reading the section guide (You're doing that now!), watching the associated videos, and completing the code exercises.
2. Submit a correct solution to the Lab exercise at the end of each chapter. These are critical as learning to code is not a spectator sport!

3. Ask an instructor to validate that you are ready for certification. If you have completed all the sections of the course program you will be certified.

Upon earning certification, your certificate will be displayed and validated by Credential.net. You'll be provided with a URL that you can use as your proof of certification. Credential.net will also allow you to link the certification to your LinkedIn account— And it is highly recommended that you do so.

*Note: As of the 2019 edition of our certification program, we are no longer requiring an online exam to earn your certification. We felt that completing the lab exercises was better proof of your programming ability than a multiple choice exam. *

Section 1 Goals

In this section of the course your goals are:

- [ ] To create your first JavaScript program
- [ ] To launch the program and see the result in your browser
- [ ] To edit and save JavaScript code
- [ ] Take Input from the user via Prompt()
- [ ] To find small errors and correct them (known as debugging) in your JavaScript code.

Watch This: Section 1 Video

As always your course videos are available on YouTube, Roku and other locations. However, only those officially enrolled have access to this course guide, are able to submit assignments, work with the instructor, and get this guide.

Watch this section video at: https://www.youtube.com/watch?v=BTEUHJh4kTY

Create and Launch Your First JavaScript Program

Since we're going to be using JavaScript mostly in the web context during this course, it makes the most sense to embed our JavaScript code in HTML so it is structured appropriately for the web browser.

We're going to use a basic HTML document structure that is common throughout the web. Every web based application or web site is based on this structure. Open your text editor and add the following code:

```
<!DOCTYPE html>
<html>
<head>
    <title>First Javascript</title>
</head>
<body>

</body>
</html>
```

This is HTML code includes the basic containers of a web based document. The head section is where you'll find metadata (data about the document) links to external styling scripts and JavaScript files and the document title tag.

In the body section is where you'll find much of the content that will be displayed to the user in the browser window. While we have our basic document structure we haven't written any JavaScript yet.

Let's add some Javascript code.

1. Place your cursor above the closing body tag which looks like this: `</body>`
2. At this point we're going to insert script tags. Between the opening and closing script tag we're going to write our actual JavaScript code. Where your cursor is flashing add the following code:

```
<script>

</script>
```

3. Now place your cursor between the opening and closing script tag and type your first line of JavaScript: `alert("Hello World from Javascript");`
4. In your text editor, you should now save your document. Choose a convenient location, like your desktop for now, so you can easily find the file. Chose a filename that ends in the extension ".html" so that your computer associates the file with your web bowers.

Your complete code file so far should look like this:

```
<!DOCTYPE html>
<html>
<head>
    <title>First Javascript</title>
</head>
<body>
    <script>
        alert("Hello World from Javascript");
    </script>
</body>
</html>
```

The alert() function is the actual JavaScript and it follows the same format as most JavaScript commands.

“Alert” is the keyword that tells the browser what to do. In this case “Alert” will create an alert box.

Between the parentheses is the function’s argument. It modifies the command. In this case our argument tells the alert command what to output within the alert box.

Now we’re ready to launch our program and see what it does in the browser.

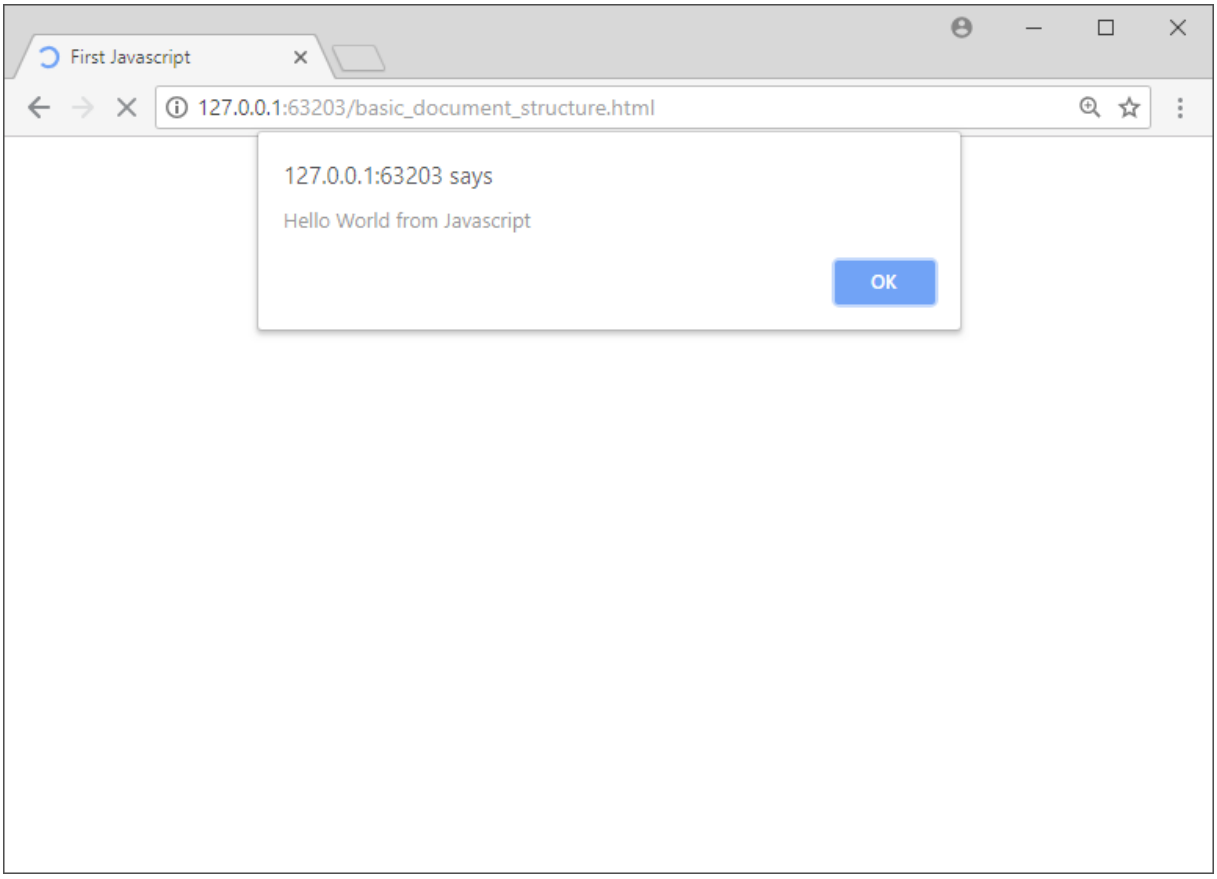
Launch Your JavaScript Program and see the Result in Your Browser

If you haven’t fired up your browser, now is the time to do so. We’re going to launch the JavaScript we just wrote in the browser so you can see the result.

There are a couple of ways to load our local file in to the browser:

1. You can double click the file on your desktop or wherever you stored it.
2. You can click the open command from your browsers menu.
3. You can drag the file in to the browser window.

Your result should look similar to the screenshot below.



The alert() command here has worked just as you’d expect, echoing the text inside the alert command argument.

Once you're done admiring your work, click "OK" and we can move on.

Edit and Save Your JavaScript Code

Frequently as a new developer you will find yourself editing existing code, not writing brand new code from scratch. To edit code your simply going to open the code in your text editor, make changes, and save the updated file. When you test your new code in the browser, the result you see will based on the changes that you have made.

Do This: Edit Your Code

To practice this process, go ahead and update your code so that instead of outputting "Hello World from JavaScript" it outputs something like "I'm coding now…" When you're successful make a few more changes to the code so that the output is different. *Practice the process of editing code, saving it and testing the output.*

Changing the Way we Output Text

Consider the following code block.

```
<!DOCTYPE html>
<html>
<head>
    <title>First getElementById</title>
</head>
<body>
    <div id="output"></div>
    <script>
        document.getElementById('output').innerHTML = "<h1>You will become a successful Javasc
    </script>
</body>
</html>
```

This is obviously more complex than the original code we created. First we have the `<div id="output"></div>` element. The purpose of this element is to give the JavaScript a place to put the output. (That's why the id is *output*.) The output will be placed inside the div versus in an alert box as before.

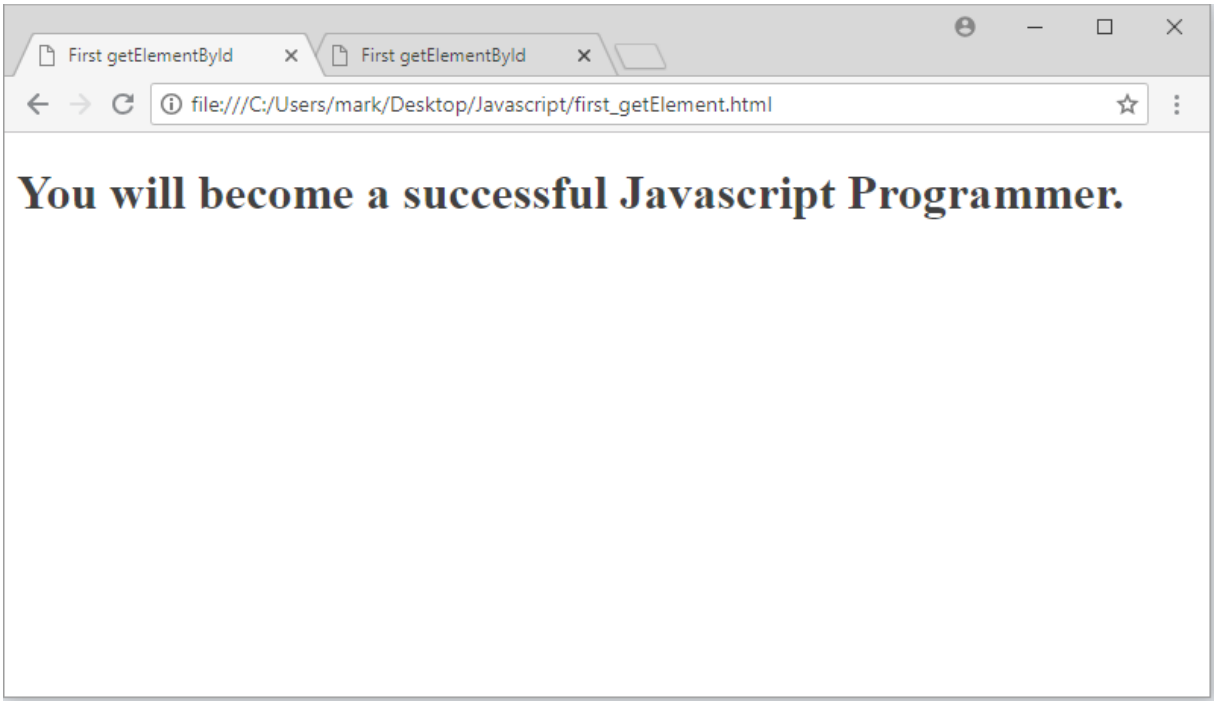
Let's see what the actual Javascript does:

document.getElementById() This command obtains an element from the HTML using the ID attribute. Notice the ID attribute in the command matches the attribute in our div.

innerHTML This is a property of the div retrieved. The property is the value of the HTML inside the div. It's blank to start, but we assign the HTML code `<h1>You will become a successful JavaScript Programmer.</h1>` This literally places the code in to the element retrieved. The code is stored in the DOM

(Document Object Model) which is a representation of the state of all possible parts of the browser.

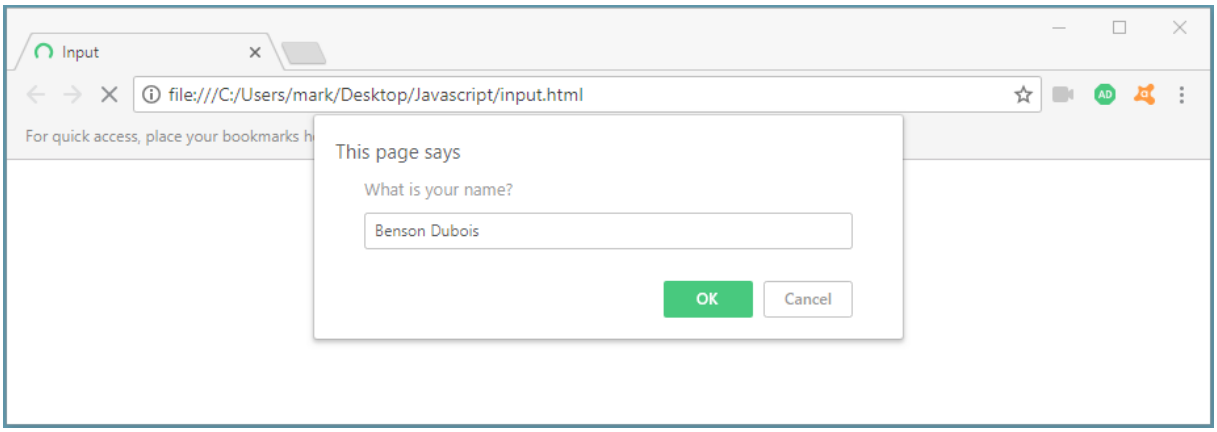
If we execute this little bit of code we get output that (hopefully) you'd expect!



Take Input from the User Via Prompt()

One more (mis)adventure for this first section. Programs become really useful when they can take input from the user and process it. While this is not Earth-shattering stuff, it does allow your programs to provide a different output based on user input.

This is done with the `prompt()` command. The prompt() command will display a pop-up which the user can fill out. It looks like the screenshot below.



Let's take a look at the code:

```
<!DOCTYPE html>
<html>
<head>
 <title>Input</title>
</head>
<body>
 <div id="output"></div>
 <script>
 var name = prompt("What is your name?");
 document.getElementById('output').innerHTML = "Greetings to " + name;
 </script>
</body>
</html>
```

As you can see the prompt() command is used just like the alert() we looked at earlier. Whatever the user types into the actual prompt stored in a variable called `name`. The greeting is inserted into the output div using the name variable which contains the name typed by the user.

In the next section, we'll be discussing the types of data that can be stored in JavaScript variables. This includes integers, floating point numbers and strings. It is important to remember that whatever input the user provides is stored in the variable as a string (series of characters) and not a number. This will become important later.

Do This: Debugging

This code has several errors. See if you can debug it and get it working as it should. When working correctly it should display a story, inserting a nouns, adjectives and verbs as indicated. As is true in the real world, these errors are not always apparent and easy to spot. Good luck.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Mad Lib</title>
    </head>
    <body>
        <div id="out"></div>
        <script>
            var properName = prompt("Enter a proper name");
            var verb = prompt("Enter a verb (past tense)");
            var verb2 = prompt("Enter a second verb (past tense)");
            var adj = prompt("Enter an adjective");
            var adj2 = prompt("Enter another adjective");

            var out = "Roses are Red.  Violets are blue. <br/>";
            out += properName + " left town without wearing a shoe. <br/>";
            out += properName + " " + verb + ".<br/>";
            out += "And " + properName + " " + verb2;
            out += "<br/>But I am " + adj + " " + properName + " thought.<br/>";
            var out += "And kind of " + adj2   " too!";

            document.getElementByID('output').innerHTML = out;

        </script>
    </body>
</html>
```

Submit This: Lab Exercise

Using everything you learned in this lesson, create your own “Mad Libs” style exercise. Your program must accept as input:

- Two Proper Names
- Three verbs
- Three Adjectives

Once the user enters the values, the program should create a story or poem similar to the debugging example above. Please save your file in the following format to insure proper credit: `LastName_Exercise1.html`.

Remember every exercise must be submitted in order for you to earn certification. Once you’ve completed this exercise, you’re ready to move on to Section II.