JavaScript’s role in Web Development

What is JavaScript (JS)?

- JavaScript (JS) is the #1 programming language for a web developer to learn.
- JS is used by millions of websites and is required for nearly all developer jobs.
- On the Frontend, JS interprets web pages as a hierarchical collection of objects, known as the **Document Object Model (DOM)**.
- On the Backend, **Node.js** and **React.js** work with a **server** object and databases.

What can JavaScript do for Websites?

Add interactivity

JavaScript enables interactivity, which empowers the user. When you click a button and something happens, that’s probably JavaScript in action.

Manipulate the DOM

JS can get any html element and do stuff, like:

- change an element’s content or modify an element’s CSS properties:
- retrieve values from form objects
- perform calculations and output the result
- When fresh data appears without the whole page reloading, that's JS.

JavaScript is written...

- inside **script** tags, in either the head or body of an html file.
- in a **.js** file, which is imported into either the head or body of an html page.

> **Per Wikipedia:**
>
> JavaScript (JS), is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. Over 97% of websites use JavaScript on the client side for web page behavior, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users' devices...
>
> JavaScript engines were originally used only in web browsers, but are now core components of some servers and a variety of applications. The most popular runtime system for this usage is Node.js ...
>
> Although Java and JavaScript are similar in name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design.

1. Launch your code editor (Visual Studio Code, Sublime Text, etc.). If you are in a Noble Desktop class, launch Visual Studio Code.

2. Navigate to the JavaScript-Fundamentals folder.

3. Go into the **00-Introduction** folder and open the file **00.01-Introduction-to-JavaScript.html**.

---

script tag

When written directly in an html page, JS is wrapped in a script tag, which goes right before the closing body tag. Putting the JS code at the end lets the web page fully load before any JS “goes looking” for any particular element to manipulate.

1. Add a pair of script tags right before the closing body tag. Separate the script tags and skip a line:

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

---

output

Programs produce output – be it a single-line greeting or a complete, interactive application. Output is also necessary at each step in the coding process, as it provides feedback that tells if the code written so far is working—or if it’s buggy (has errors).

---

alert()

alert() is a JS method. Think of methods as the verbs of programming languages. They perform actions. In this case, the action is to pop-up a dialog box that displays the string (text) which was passed to the parentheses.

1. Inside the script tags, call the alert() method:

```
<script>
  alert();
</script>
```

---

string

In JS and other programming languages, plain text is known as a string. Strings go in quotes, either single or double.

1. Inside the script tag, add an alert containing the string 'Hello World':

```
<script>
  alert('Hello World');
</script>
```

2. Save the file and open it in Chrome. You should get an alert (browser pop-up) that says Hello World.

3. Close the alert, but leave the html page open in the browser. We’ll be reloading the page often as we proceed.

console.log()

Running Javascript in the Browser Console:

- You do not need a DOM, that is to say, a web page, in order to write and run Javascript.
- You can write JS directly in the **Console** for testing purpose, as many developers do—-no html page required.
- To output content to the Console, use **console.log()**.

1. Open the Console. In the Chrome browser, right click the page and choose **Inspect**. Then click the **Console** tab. The keyboard shortcut for opening the Console is **command-option-i**.

Now, we'll switch to the console.log() method, which is what we'll be using from now on. console.log() also outputs whatever is in its parentheses, but its output goes to the Console.

1. Use console.log() to output another message. Strings (text) can go in either double or single quotes, so switch to double quotes:

   ```
   alert('Hello World');
   console.log("Hello from the Console");
   ```

2. Save the file and reload the browser. You get the alert again, but there's more:
3. Right-click the page and choose Inspect; then click the Console tab. It should say: Hello from the Console

Global Window Object

The **global window object** is the top-level object in JavaScript, corresponding to the browser window itself.

- **alert()** is a method of the window
- **console** is an child object of the *window* object.
- **log()** is a method of the *console* object. Because the window object is top-level, it is assumed, and therefore can be omitted from the syntax. But we can also add them:

1. Put **window** in front of the alert and console keywords and run it again. It still works:

   ```
   window.alert('Hello World');
   window.console.log("Hello from the Console");
   ```

semi-colon

Notice the semi-colons (;) which mark the end of each line of code. Semi-colons are optional, but recommended. We'll be using them.

comments

Comments explain what the code is doing. Comments are for us humans and, as such, are ignored by JS when the program is run. While comments have no effect on how a program runs, they do make the code easier to read and understand. We recommend you comment your code. Even if no one else will be reading the code, it will help you remember what you wrote and why later.

---

single-line comment

A // at the start of a line turns the whole line into a comment.

14. Add a couple of one-line comments to start the script:

```
// alert() and console.log() output what’s in parentheses
// window. before alert and console are optional
window.alert('Hello World');
window.console.log("Hello from the Console");
```

---

in-line comment

If the code and its comment are short enough, the comment can go on the same line as an in-line comment.

15. Add an in-line comment to describe an alert():

```
window.alert('Hello World'); // browser pop up
window.console.log("Hello from the Console");
```

---

commenting-out code

Comments are used to deactivate code without having to delete it. A double slash at the start of a line comments it out. By commenting out code instead of deleting it, we keep it around for study and review purposes or in case we want to reactivate it later. The keyboard shortcut for commenting out code is **command-slash**.

16. Comment-out the alert so that we don't keep getting that pop-up.

```
// window.alert('Hello World'); // browser pop-up
window.console.log("Hello from the Console");
```

---

multi-line comments

If a comment runs multiple lines, you can wrap all in /* ... */.

17. Replace the single-line comments with a multi-line comment; keep the commented-out alert for study and reference:

```
/*
Ways JS outputs content (besides to a Web page)
alert("howdy") -- gives a pop-up that says howdy
console.log("hola") -- outputs hola to the Console
Global Window Object: the top level object in JS
alert() is a method of window object
console is a child object of window object
Therefore, you may optionally begin w window-dot:
window.alert("howdy")
window.console.log("hola")
*/

// window.alert('Hello World'); // browser pop-up
window.console.log("Hello from the Console");
```