JavaScript

JavaScript is one of the most used languages in the world. It is mostly used for web development and mobile development. `JavaScript` is usually used on the front end of an application to be executed within a browser. Still, there are implementations of back end JavaScript used to develop entire web applications, like NodeJS.

While `HTML` and `CSS` are mainly in charge of how a web page looks, `JavaScript` is usually used to control any functionality that the front end web page requires. Without `JavaScript`, a web page would be mostly static and would not have much functionality or interactive elements.

Example

Within the page source code, `JavaScript` code is loaded with the `<script>` tag, as follows:

Code: html

```
<script type="text/javascript">
..JavaScript code..
</script>
```

A web page can also load remote `JavaScript` code with `src` and the script's link, as follows:

Code: html

```
<script src="./script.js"></script>
```

An example of basic use of `JavaScript` within a web page is the following:

Code: javascript

```
document.getElementById("button1").innerHTML = "Changed Text!";
```

The above example changes the content of the `button1` HTML element. From here on, there are many more advanced uses of `JavaScript` on a web page. The following shows an example of what the above `JavaScript` code would do when linked to a button click:

Original Text

As with HTML, there are many sites available online to experiment with `JavaScript`. One example is JSFiddle which can be used to test `JavaScript`, `CSS`, and `HTML` and save code snippets. `JavaScript` is an advanced language, and its syntax is not as simple as `HTML` or `CSS`.

Usage

Most common web applications heavily rely on `JavaScript` to drive all needed functionality on the web page, like updating the web page view in real-time, dynamically updating content in real-time, accepting and processing user input, and many other potential functionalities.

`JavaScript` is also used to automate complex processes and perform HTTP requests to interact with the back end components and send and retrieve data, through technologies like Ajax.

In addition to automation, `JavaScript` is also often used alongside `CSS`, as previously mentioned, to drive advanced animations that would not be possible with `CSS` alone. Whenever we visit an interactive and dynamic web page that uses many advanced and visually appealing animations, we are seeing the result of active `JavaScript` code running on our browser.

All modern web browsers are equipped with `JavaScript` engines that can execute `JavaScript` code on the client-side without relying on the back end webserver to update the page. This makes using `JavaScript` a very fast way to achieve a large number of processes quickly.

Frameworks

As web applications become more advanced, it may be inefficient to use pure `JavaScript` to develop an entire web application from scratch. This is why a host of `JavaScript` frameworks have been introduced to improve the experience of web application development.

These platforms introduce libraries that make it very simple to re-create advanced functionalities, like user login and user registration, and they introduce new technologies based on existing ones, like the use of dynamically changing `HTML` code, instead of using static `HTML` code.

These platforms either use `JavaScript` as their programming language or use an implementation of `JavaScript` that compiles its code into `JavaScript` code.

Some of the most common front end `JavaScript` frameworks are:

- `AngularJS`
- `React.js`
- `Vue.js`
- `jQuery`

A listing and comparison of common JavaScript frameworks can be found here.

Previous Next

Mark Complete & Next

Table of Contents

Introduction to Web Applications
Introduction
Web Application Layout
Front End vs. Back End

Front End Components
HTML
Cascading Style Sheets (CSS)
JavaScript

Front End Vulnerabilities
Sensitive Data Exposure
HTML Injection
Cross-Site Scripting (XSS)
Cross-Site Request Forgery (CSRF)

Back End Components
Back End Servers
Web Servers
Databases
Development Frameworks & APIs

Back End Vulnerabilities
Common Web Vulnerabilities
Public Vulnerabilities

Next Steps
Next Steps