Writing JavaScript for the Web

by Devmountain Tutorials

WHAT'S COVERED

This section will explore how to write JavaScript for the web by discussing:

1. WEB APIs
2. THE DOM
3. THE EVENT SYSTEM

1. WEB APIs

The real power of JavaScript on the web comes from its ability to give web developers access to browser APIs. Earlier, you learned that APIs are pre-built pieces of code that you can repurpose in your own code. Web browsers come with a collection of APIs known as browser APIs. The subset of browser APIs used for writing code for the web are known as Web APIs. There are a lot of **Web APIs**; we won't go over all of them in this section but **Mozilla Developers' Network has a list of all available Web APIs** with links to pages where you can learn more about each one.

This section will focus on the most commonly used Web API — the **Document Object Model (DOM)** — an API that gives you programmatic access to the contents, style, and structure of an HTML page. It's the API used to dynamically change the contents of a web page which allowed developers to program complex behaviors like ones found in web applications.

2. THE DOM

Arguably the most commonly used Web API is the **Document Object Model (DOM)**. The DOM allows you to access and manipulate the structure of an HTML document by representing it as objects in memory, where the objects are accessible via JavaScript. In other words, it's a model of an HTML document where elements are represented as JavaScript **objects**. You can access these objects in order to change the web page's structure, style, or content.

Here's an example of changing a web page's structure. Consider the following HTML:

```
<h1>Hello,</h1>
<p>world!</p>
```

element appears before the **p** element, **Hello**, will appear before **world!**. With JavaScript, we can mess with the natural order of things so **world!** appears before **Hello,**:

The DOM can also be used to generate content. The list items in the example below are generated using JavaScript. The only HTML present on the page is an empty **ul**element.

These examples are a bit contrived though. After all, if someone *really* wanted

world!
to appear before
<h1>Hello,</h1>
they could just write their HTML that way to begin with and save themselves the trouble of adding any JavaScript! The real power that programmatic access to an HTML document gives you comes from being able to change the contents of a page dynamically. For example, you can change the page based on certain conditions.
Here's an example that will display a different message depending on your local time. If you're reading this before noon, the paragraph will say **Good morning!**, otherwise, it'll say **Good evening (or afternoon)**!

The DOM also gives you access to the browser's event system, which you can leverage to trigger code based on events like clicking on a button or scrolling past a certain area on the page. We'll cover the event system in more detail next.

TERM TO KNOW

Document Object Model (DOM)
An API that gives programmatic access to the contents, style, and structure of an HTML page. It is used to dynamically change a web page's contents.

Object
In programming, the term object is equivalent with value. For example, the value 100 can be described as a number object.

3. THE EVENT SYSTEM

The browser's event system is based on a programming design pattern called**event-driven programming** where the flow of a program is determined by events like mouse clicks, key presses, or even messages from other programs. The system consists of **event listeners** that listen for certain events and event handlers that execute code when certain events occur.

Using the DOM, you can add an event listener to an HTML element using a special function called **addEventListener**. **addEventListener** needs two inputs — the name of an event and a function containing the code that should be triggered when the event occurs. For example, here's a button that adds items to a list whenever it's clicked:

Pay close attention to the last lines of code:

const listAdderButton = document.querySelector('button');

tListener('click', addItemToList);

In the code above, **listAdderButton.addEventListener** adds an event listener to the **button** element. **'click'** is the type of event — a mouse click — and **addItemToList** is a variable whose value is a function that generates an **li** element and adds it to the list.

To add an event listener to an HTML element, you need three values:

- The HTML element itself
- The type of event you want to listen for, as a string
- A function that contains the code you want to execute whenever the event occurs

The code below has a text box and a paragraph that should display the number of words that the user has entered into the text box. Follow the steps to add code to the Editable code area below to add an event listener to the text box so that it'll update the word count whenever the user types into the text box. Remember, *Reset* will reset everything and *Show Solution* will show the solution.

TRY IT

Follow these steps in the box below:

1. Add this line at the bottom to declare a variable and set its value to the text box:**const textbox = document.querySelector('#textbox');**

2. Press the *Enter* key to start a new line. Then, type **textbox.addEventListener** and an opening parenthesis **(**

3. Next, list the two inputs that **addEventListener** needs:
   - Type **keyup**. This event occurs whenever a key has been pressed. Then, add a comma**,** and a space.
   - Type **updateWordCount**, which is the name of the function that updates the word count.

4. Close everything up with a closing parenthesis **)** and a semicolon **;**.

We've covered many concepts in this unit! It probably still feels overwhelming, and that's okay. The more you experiment with code, the easier it will be. If this intrigued you, we encourage you to keep learning. There are many online resources, college courses, or even full bootcamp programs like Devmountain offers. In the next, and final unit, we'll introduce you to some computer engineering concepts that many web developers leverage.

TERMS TO KNOW

Document Object Model (DOM)
An API that gives programmatic access to the contents, style, and structure of an HTML page. It is used to dynamically change a web page's contents.

Object
In programming, the term object is equivalent with value. For example, the value 100 can be described as a number object.