

Programming for the Web

by Devmountain Tutorials



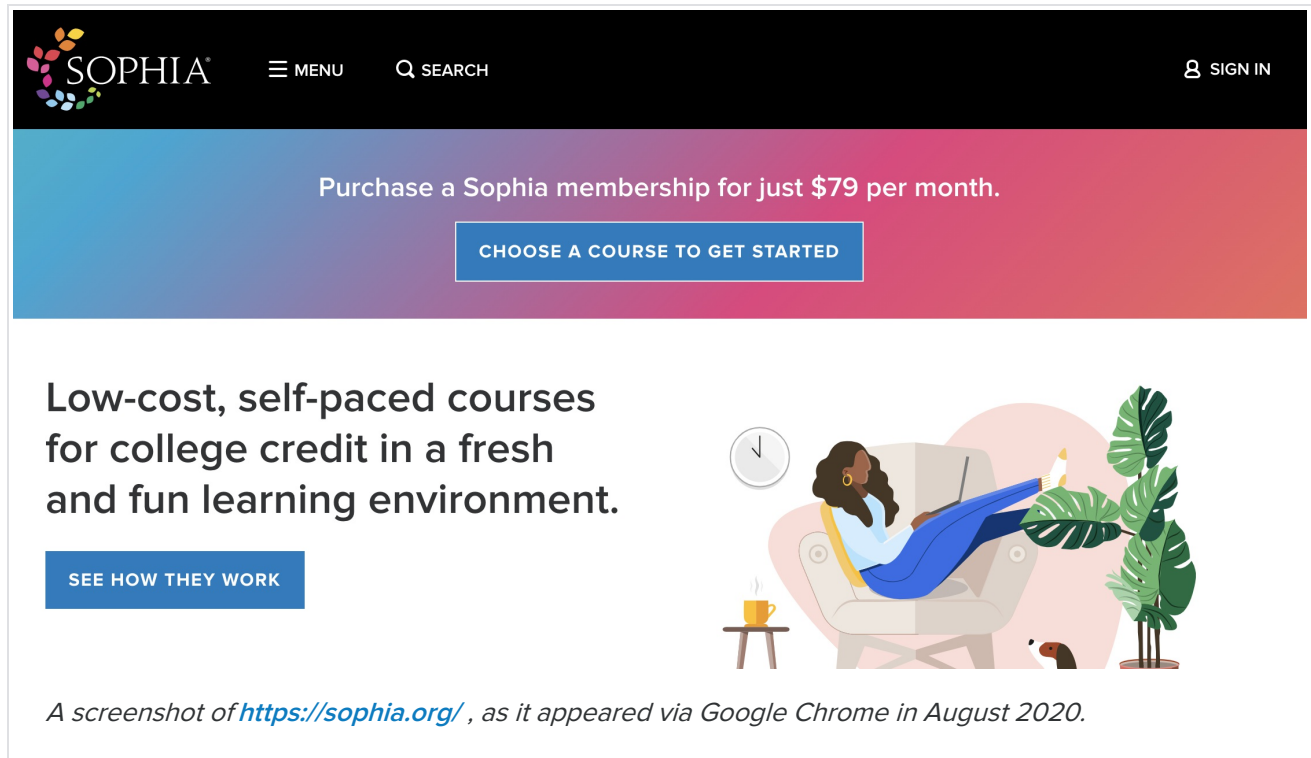
WHAT'S COVERED

This section will explore how to use programming languages to create a web page by discussing:

1. WHAT'S IN A WEB PAGE?
2. HTML
3. CSS
4. JAVASCRIPT

1. WHAT'S IN A WEB PAGE?

At the beginning of this unit, you learned that a web page is a collection of information that you can access from a web browser. Every web page has a distinct URL. You've probably entered a web page's URL into your browser whenever you wanted to view the contents of that particular page. For example, <https://sophia.org/> takes us to a web page that looks like this:



The screenshot shows the Sophia Learning website. At the top is a dark navigation bar with the Sophia logo on the left, a 'MENU' button in the center, a 'SEARCH' button on the right, and a 'SIGN IN' link on the far right. Below the navigation bar is a large banner with a blue-to-pink gradient. The banner contains the text 'Purchase a Sophia membership for just \$79 per month.' and a blue button labeled 'CHOOSE A COURSE TO GET STARTED'. Below the banner, on the left, is the text 'Low-cost, self-paced courses for college credit in a fresh and fun learning environment.' with a blue button labeled 'SEE HOW THEY WORK'. On the right side of this section is an illustration of a woman with long dark hair sitting in a modern armchair, reading a book. A small table next to her holds a steaming cup of coffee. A clock on the wall shows the time is around 1:50. A potted plant is to the right of the chair.

A screenshot of <https://sophia.org/> , as it appeared via Google Chrome in August 2020.

All web pages like the one above are made up of three different languages — HTML, CSS, and JavaScript. Each language is responsible for a different aspect of a page; HTML defines the structure of a page's content, CSS provides rules for a page's look and feel, and JavaScript is used to program things like animations and interactivity. Let's talk about each language in more detail.

2. HTML

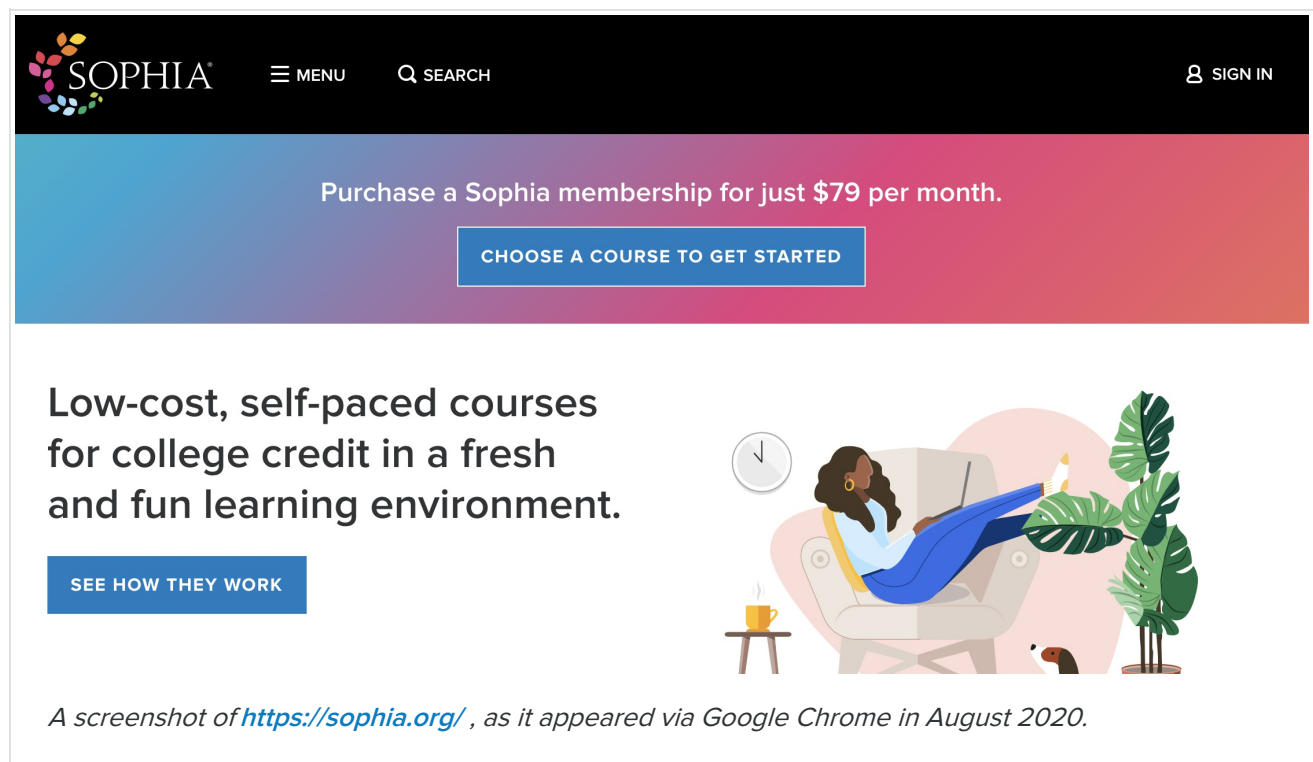
HTML stands for **HyperText Markup Language**. As you might have gathered from its name, HTML is a **markup language**. Like programming languages, markup languages are a type of code. Unlike programming languages, markup languages aren't used to write algorithms. Instead, they're used to group, annotate, and categorize content to describe the content's semantics in a way that computers will understand.



DID YOU KNOW

Other markup languages include reStructuredText, Markdown, and AsciiDoc.

For example, here's that screenshot of the Sophia home page again:



As humans, it's easy for us to understand the structure of this page. We have the benefit of being able to use contextual clues and our past experiences of navigating similar web pages to analyze the page and make sense of it. For instance, we can tell that the black bar at the top is used for navigation because it's a design pattern employed in many other web pages. We can also assume that the color-shifting box below the navigation bar is a special announcement because its bold background makes it distinct from the rest of the page.

Computers, on the other hand, can't determine any of those things without our help. This is what the Sophia home page might look like to a computer if HTML didn't exist:

sophia-logo.png Menu Search Sign In Purchase a Sophia membership for just \$79 per month. Choose a course to get started Low-cost, self-paced courses for college credit in a fresh and fun learning environment. See how they work woman-in-chair.jpg

Even humans will have trouble making sense of the content above! It's much harder to tell where the navigation bar starts and ends. Also, it's nearly impossible to distinguish between the content that's part of the special announcement and the content that's outside of it.

Now, compare the text above to the text below. It contains the exact same content but, this time, we've added some HTML:

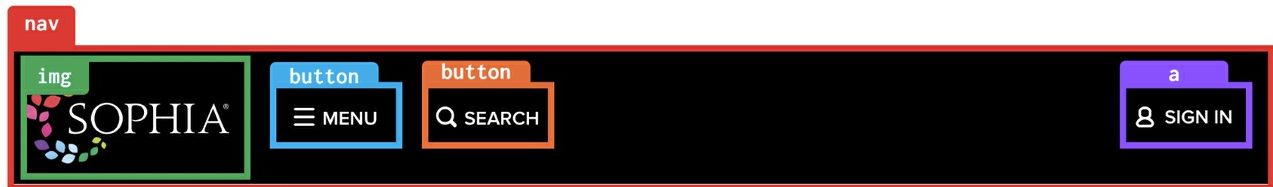
```
<nav>
  
  <button>Menu</button>
  <button>Search</button>
  <a href="/sign-in">Sign In</a>
</nav>
<aside class="announcement">
  <p>
    Purchase a Sophia membership for just $79
    per month.
  </p>
  <button>Choose a course to get started</button>
</aside>
<main>
  <section id="about-courses">
    <div class="left">
      <p>
        Low-cost, self-paced courses for college credit in
        a fresh and fun learning environment.
      </p>
      <button>See how they work</button>
    </div>
    <div class="right">
      
    </div>
  </section>
</main>
```

Even without a complete understanding of HTML, it's a lot easier to tell that certain parts of the page are associated with one another. HTML allows developers to tell the computer that, for example, all the content between `<nav></nav>` represent the section of the page whose purpose is to provide navigation links between different web pages and that `Menu` and `Search` aren't just random words, they're buttons.

```

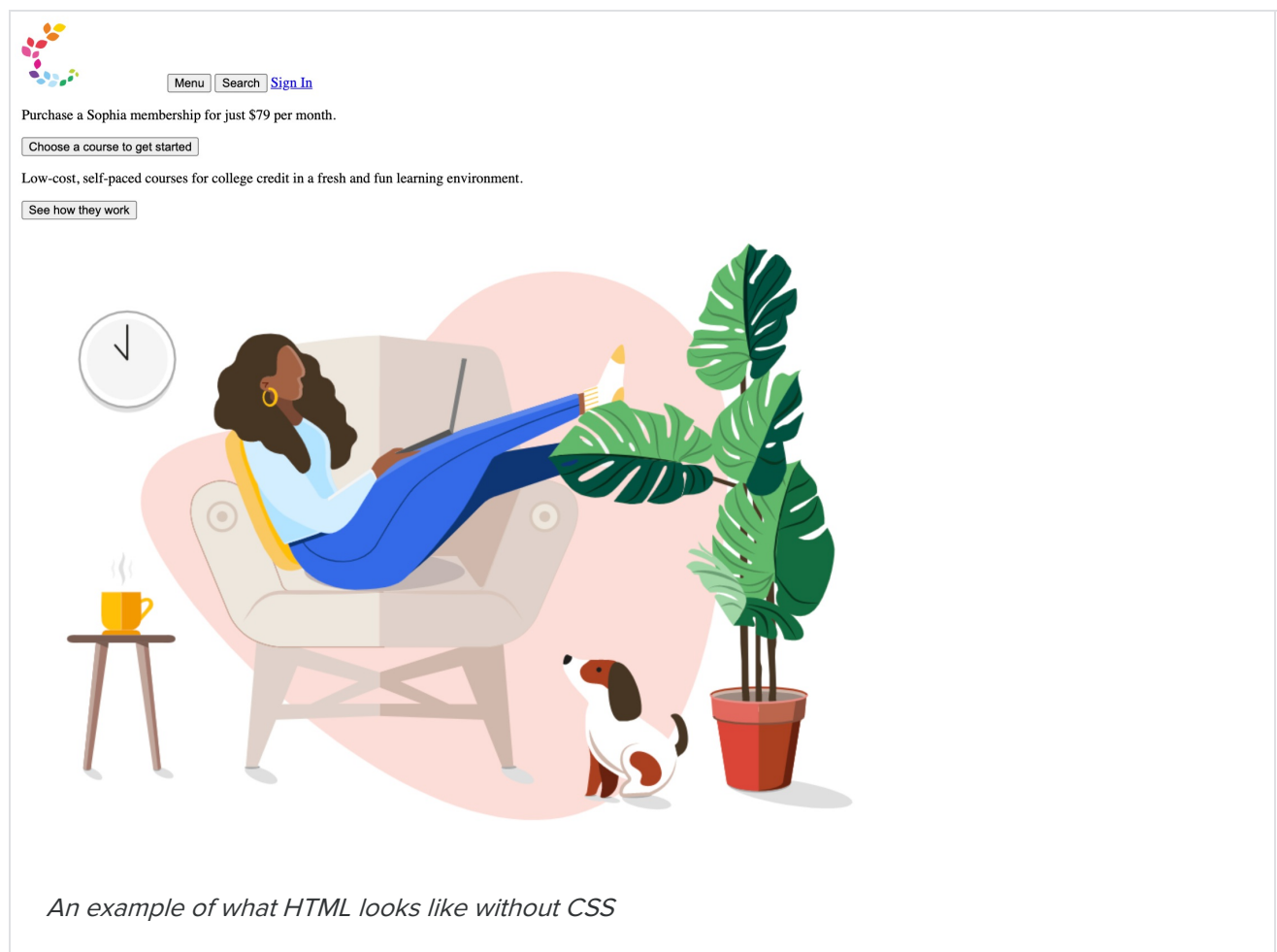
<nav>
  
  <button>Menu</button>
  <button>Search</button>
  <a href="/sign-in">Sign In</a>
</nav>

```



Graphical user interface, text.

HTML isn't the only language at work though. Without CSS and JavaScript, Sophia's homepage would look something like this:



An example of what HTML looks like without CSS

Notice how, in the screenshot above, we can see all the content of <https://sophia.org/> but it doesn't look right. The fonts are wrong, the navigation menu doesn't have a black background, and the buttons look boring! HTML is just one of the components that make up a webpage and it's only responsible for determining the

contents of a page. To make those contents look stylish and visually appealing, you'll need a language that'll specify how documents are presented. That's where CSS comes in.



TERM TO KNOW

HyperText Markup Language (HTML)

A markup language that is a type of code used to group, annotate, and categorize content in a way that computers understand.

3. CSS

CSS (Cascading Style Sheets) is used to specify how a document should be presented to users. Here, **presenting** a document to a user means converting it in a way that's usable by your audience. When a web browser presents a document to a user, it converts HTML source code into a visual format that can be displayed on a computer screen.

CSS has many capabilities. It can be used for basic text styling like changing the font, color, size, and weight of headings, paragraphs, and links.

It's also used to create the layout of a page so that content is displayed across two columns instead of just one.

CSS can do complex things too like drawing shapes, adding animations, adding effects when users click on a certain element, and more. CSS has so many features that web developers are even able to make complex illustrations using CSS and a bit of HTML.

CSS is good for making visual changes to a page and some dynamic behavior, but if you want more control over the contents of a page so that you can dynamically update content, respond to user input like button clicks, and nearly everything else you'd want to do on a website, you'll want to use JavaScript.



TERM TO KNOW

CSS (Cascading Style Sheets)

Specifies how a document should be presented to users.

Presenting

Converting a document in a way that is usable by your audience. When a web browser presents a document to a user, it converts HTML source code into a visual format that can be displayed on a computer screen.

4. JAVASCRIPT

You're already familiar with JavaScript's history, how monumental it was for accelerating the growth of the web, and that it's a programming language. Now you'll see how it builds on top of HTML and CSS. Let's start small with some HTML:

It's a bit bland though, so let's spice it up with CSS:

Let's make this more interactive by adding JavaScript so that users can get a customized greeting when they click on **Hello, friend!**:

By the way, the example above is interactive. Try clicking on **Hello, friend!** and see what happens.

The example above illustrates how JavaScript is generally used on the web. It can store data (like when we asked for a name and stored it in a variable called **name**), it can perform operations on numbers and text, and it can execute code in response to user interaction.

What's even better is how you can use JavaScript to take advantage of **Application Programming Interfaces (APIs)**, ready-to-use sets of code building blocks that you can repurpose in your own project. Think of them like furniture kits that come with panels of wood already cut and sanded, and all you have to do is put the pieces together. It's much easier to do that than to build a chair, completely from scratch. APIs used on the web (APIs exist in all types of software engineering) are typically **browser APIs** or **third-party APIs**.

Browser APIs come built into your web browser. You can use them to hook into your computer's environment or do all sorts of complex things. For example, in the examples above, we used a browser API to manipulate HTML. There's an API that you can use for geolocation and geographical information and there are APIs for creating 2D and 3D graphics.

Third-party APIs don't come with your browser so you have to acquire or install their code separately. Many companies publish APIs that allow developers to access their data. For example, Google has an API that allows you to use Google Maps' location data and traffic information. Spotify has an API you can use to look up musicians, albums, and songs. There are a lot of cool APIs available to use for free, thanks to the fact that software engineers have established a culture of freely sharing knowledge and code with each other.

We hope you're excited about the endless possibilities of programming on the web. You now have all the pieces needed to understand the roles that HTML, CSS, and JavaScript play in creating a website. These three languages are arguably a web developer's most powerful tools, but there are peripheral tools besides just coding languages that developers use on a daily basis. Some might even argue that they're more important than HTML, CSS, and JavaScript! In the next section, we'll talk about those peripheral tools — ones that web developers use to write and run code.



TERM TO KNOW

Application Programming Interfaces (APIs)

Ready-to-use sets of code building blocks that you can repurpose in your own project.



TERMS TO KNOW

Application Programming Interfaces (APIs)

Ready-to-use sets of code building blocks that you can repurpose in your own project.

CSS (Cascading Style Sheets)

Specifies how a document should be presented to users.

HyperText Markup Language (HTML)

A markup language that is a type of code used to group, annotate, and categorize content in a way that computers understand.

Presenting

Converting a document in a way that is usable by your audience. When a web browser presents a document to a user, it converts HTML source code into a visual format that can be displayed on a computer screen.