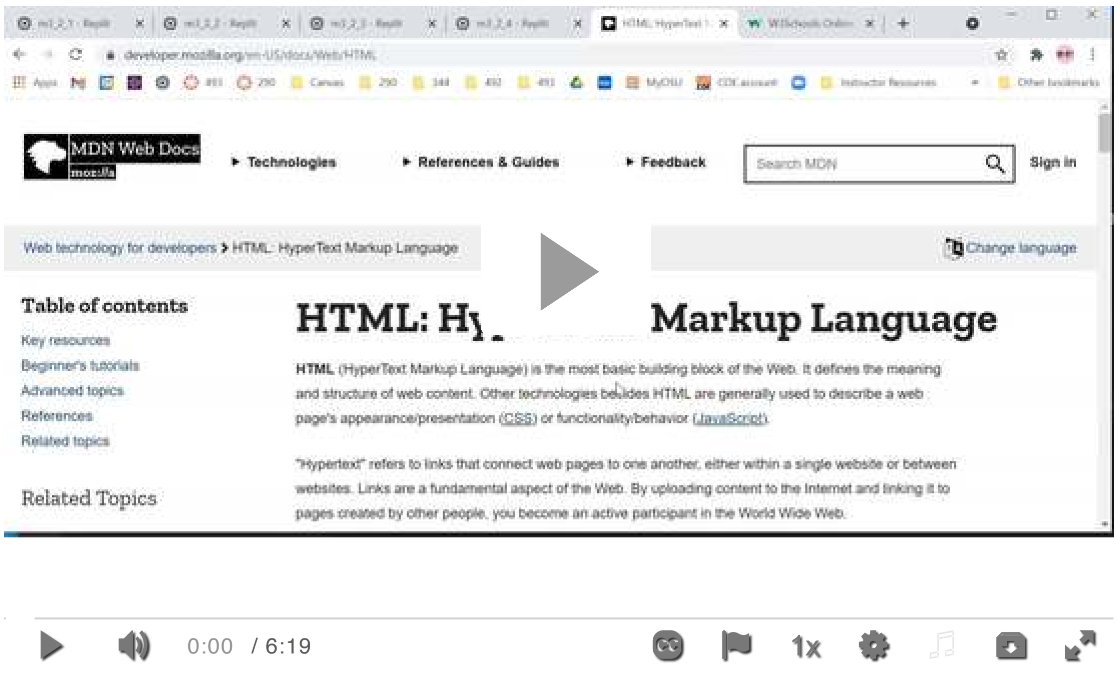
Exploration — Introduction to HTML

Introduction

HTML is one of the foundational technologies for the web. It stands for HyperText Markup Language and is the language used to describe the structure and meaning of webpages. Breaking this down, HyperText refers to documents that link to one another. This is one of the defining attributes of HTML, i.e., we one can navigate from one HTML document to another HTML document. The other piece is that it is a Markup Language. This means that it is basically a language to describe webpages. This is distinct from a programming language that allows for the use of logic and control structures. The current version of HTML is **HTML5 (https://en.wikipedia.org/wiki/HTML5)** and the **standard is available here (https://html.spec.whatwg.org/)**.



HTML: Hy Markup Language

HTML (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/behavior (JavaScript).

"Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web. By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.

Elements and Tags

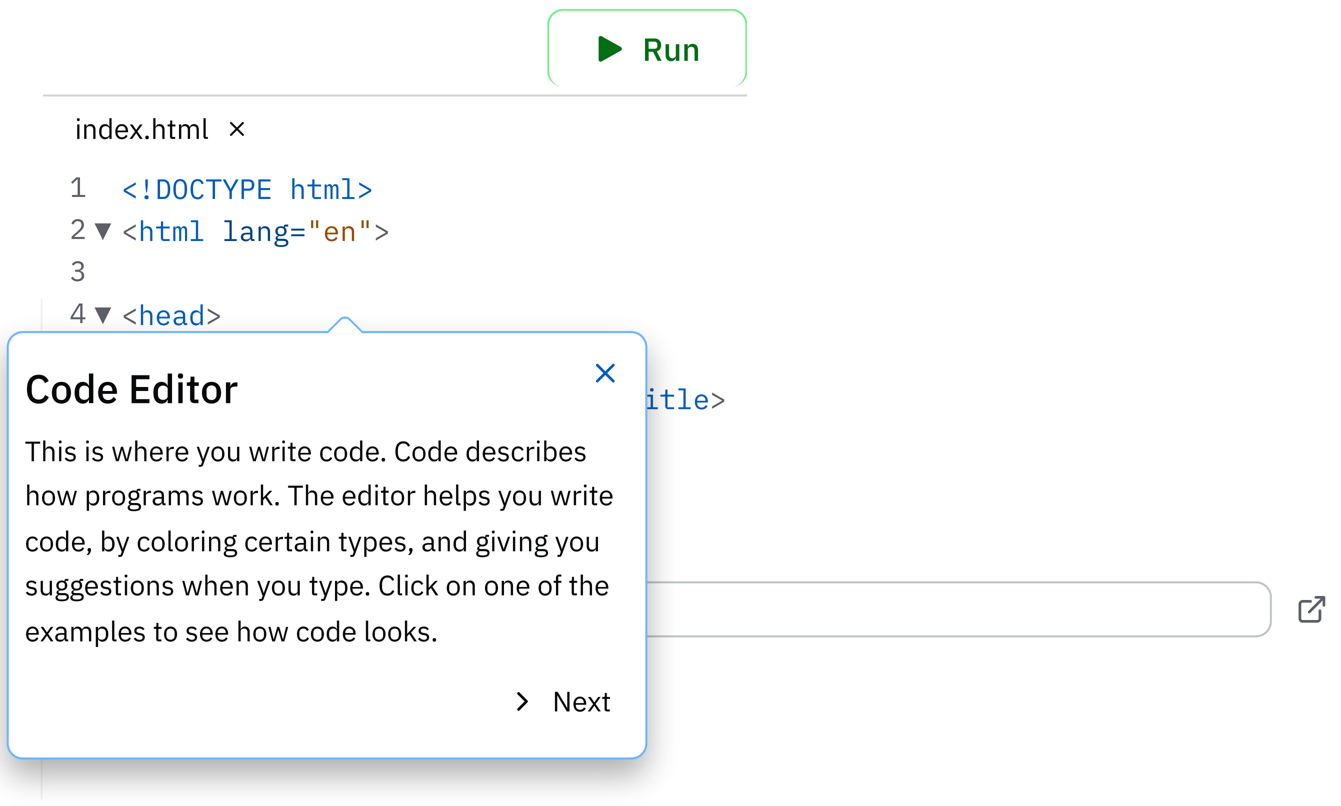
HTML structures documents by using **elements**. Elements can stand on their own or be nested within other elements. For example, a paragraph element might exist within a section element or a table row will be within a table. The way elements are represented in HTML is by using a pair of opening and closing tags. An opening tag has the element name enclosed in `<>` and a closing tag has the element name enclosed in `</>`.

Note: There are a few elements that have just one tag (it self-closes), rather than a pair of tags. For example: `<img />` (denotes an image file). These elements cannot have any other elements nested in them and are called **empty elements** **(https://developer.mozilla.org/en-US/docs/Glossary/Empty_element)**. We will encounter of some of these elements within this exploration and this module.

Example

The HTML element `<h1>` is used to mark the enclosing text as a section level heading. It is one of 6 section level headings that range from `<h1>` to `<h6>`, with `<h1>` being the highest level and `<h6>` the lowest level. They are used hierarchically, just like the numerals of an outline.

```
<h1>Display this as the highest level heading.</h1>
```



```
<!DOCTYPE html>
<html lang="en">

<head>
    itle>
```

Code Editor

This is where you write code. Code describes how programs work. The editor helps you write code, by coloring certain types, and giving you suggestions when you type. Click on one of the examples to see how code looks.

Structure of an HTML Document

Here is the most basic structure of an HTML document

```
<!doctype html>
<html lang="en">
  <head>
        <meta charset="utf-8">
        <title>Basic Structure of an HTML Document</title>
  </head>
  <body>
  </body>
</html>
```

The first line `<!doctype html>` is called the **Document Type Declaration**. This tells the browser that this is an HTML document. Following this, we have the `<html>` element which is the root element of the document. This element typically has a `lang` attribute to describe the language used by the majority of content on the page. All other elements are descendants of this element. Inside the `<html>` element, we have the elements `<head>` and `<body>`.

The `<head>` element is used to specify meta-data about the document and is also called the **header of the document** (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/head). It helps the browser and robots (e.g., search engine crawlers) understand what is on the page. In the above example, we specify the character set being used and provide a title for the document. If we open this document in a browser, the **browser tab** title will show the contents of the `<title>` element.

The `<body>` element contains the content which is visible in the browser's viewport (more about that in Module 4), below the tab and address bar.

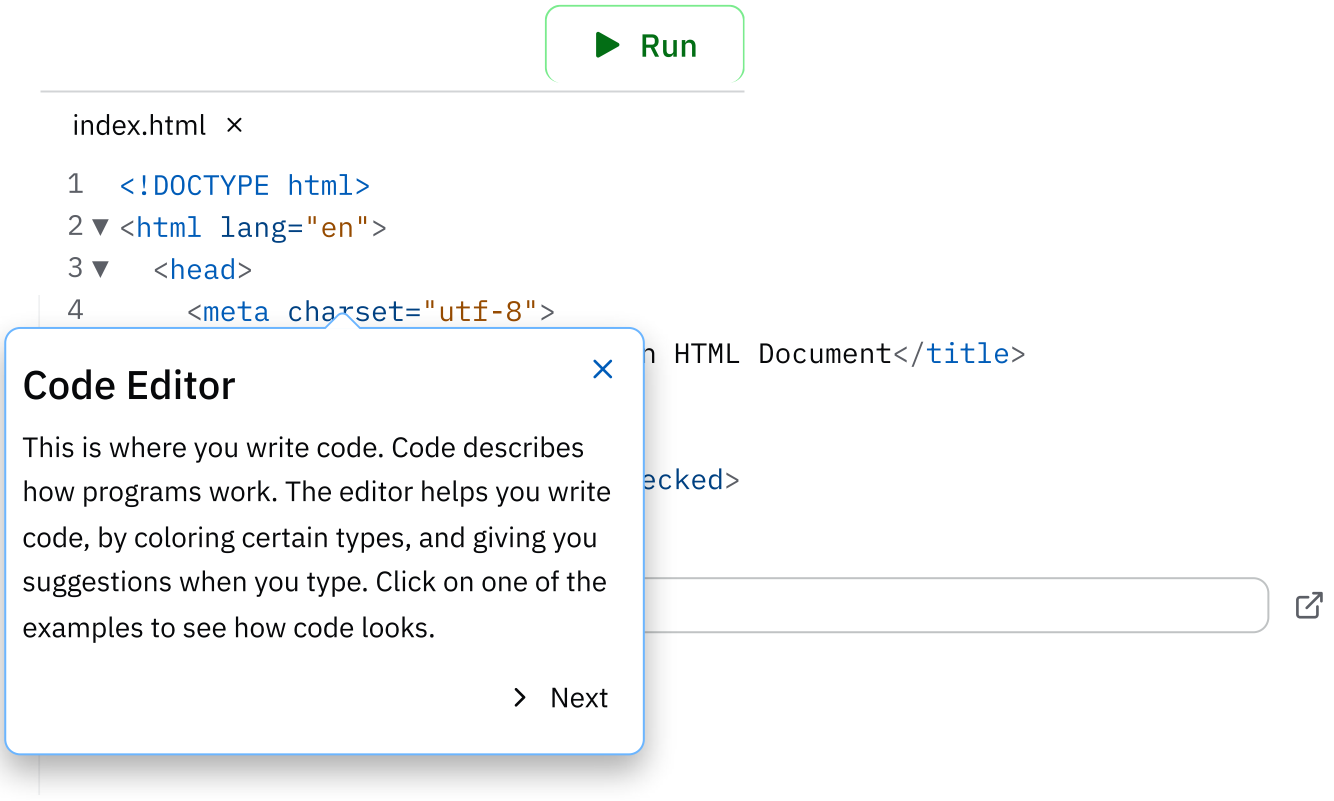
The HTML tags are not case sensitive, so `<BODY>` is the same as `<body>`. However, HTML 5 specifications expect developers to use lowercase elements (and it will make your markup of content easier).

Attributes

In addition to basic opening and closing tags, additional information can be provided to those tags using **attributes**. Attributes are usually specified using the syntax: `attribute="value"` with either single or double quotation marks.* These pairs must be separated by a space. On occasion, you will see an attribute name with no associated value.

* It is common for developers to use double quotation marks in HTML and single quotes in scripts.

Example



```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
```

Code Editor

This is where you write code. Code describes how programs work. The editor helps you write code, by coloring certain types, and giving you suggestions when you type. Click on one of the examples to see how code looks.

The HTML element `<input>` is used to create controls for user input. Note that `<input>` is an empty element (self-closing). In the above example, we have specified two attributes for this element.

```
<input type="checkbox" checked>
```

- The `type` attribute of this element determines the behavior of this element. `type="checkbox"` means that the element is a checkbox that can be either selected or deselected.
- The `checked` attribute which doesn't have a value. It denotes a default value. When we specify a checked checkbox, it will appear as checked/selected in the webpage. The user can uncheck it and check another.

Exercise

Edit the above example to remove the attribute `checked` and click "Run". How does the displayed HTML page change?

Nesting Elements

Opening and closing tags act a lot like parenthesis. If a tag has both an opening and closing tag, those must both appear at the right depth inside the tree of elements. For example, you could have two paragraphs, one of which has emphasized text, inside an article. This is the **valid, nested** way to mark-up those elements:

```
<article>
   <p>I am paragraph 1.</p>
   <p>I <em>am</em> paragraph 2.</p>
</article>
```

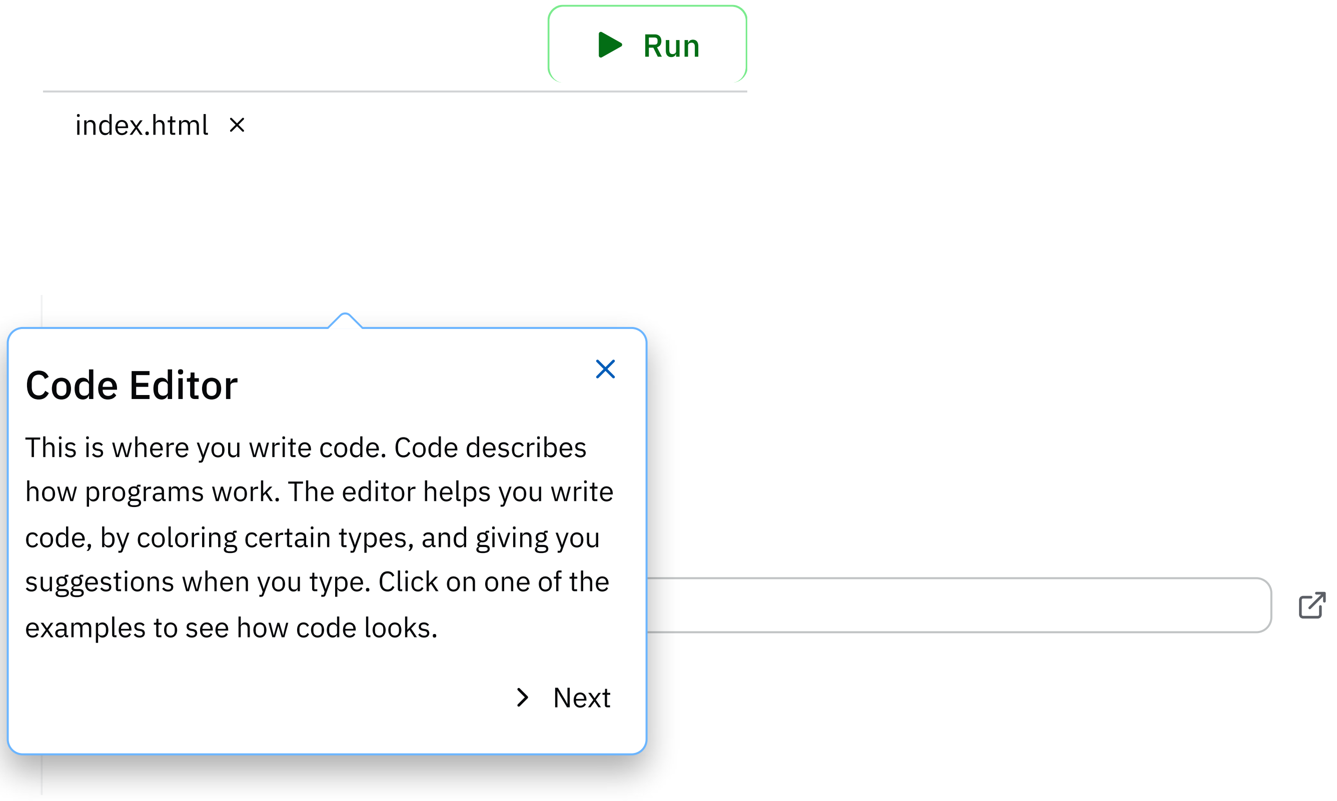
However, the following is **not valid** because the emphasized text *opens inside of the paragraph but closes outside of it:*

```
<article>
   <p>I am paragraph 1.</p>
   <p>I <em>am paragraph 2.</p></em>
</article>
```

But browsers are very forgiving of HTML errors. A lot of invalid HTML documents are rendered seemingly correctly by modern browsers. But we should strive to eliminate HTML errors instead of depending on browsers to correctly handle the errors. We can use tools, such as validators and linters, for this purpose. There are many online validators, such as the **W3C Unicorn Validator, (https://validator.w3.org/unicorn/)** which can scan an HTML document and identify warnings and errors. Similarly, many text editors and IDEs have "linters" that can highlight issues with HTML syntax. For example, the **extension HTMLHint (https://marketplace.visualstudio.com/items?itemName=mkaufman.HTMLHint)** for Visual Studio Code can analyze HTML documents, highlights issues, and provide tips for fixing the issues.

Exercise

The following HTML document has some issues. Fix the tags so they are valid. In addition, we can optionally replace the quotation marks with opening and closing `<q>` tags because they represent actual quotes. Finally, fix the emphasis tag, `<em>` so that it only emphasizes the text *that*.



Code Editor

This is where you write code. Code describes how programs work. The editor helps you write code, by coloring certain types, and giving you suggestions when you type. Click on one of the examples to see how code looks.

Using Entities to Display Special Characters in HTML Content

Certain characters in HTML are reserved for special use. For example, any text between `<` and `>` is interpreted as the name of an element. We can display a literal `<` or `>` in a document by using **entities**. Entities are strings (https://developer.mozilla.org/en-US/docs/Glossary/Entity) that begin with `&` and end with `;` and are provided to display reserved characters as well as characters that may be difficult to type on a keyboard. Here are some common entities:

Entity	**Description**
`&lt;`	Displays `<`
`&gt;`	Displays `>`
`&amp;`	Displays `&`
`&quot;`	Displays `"`
`&copy;`	Displays `©`

Summary

This should get you familiar with the syntax and various pieces of HTML tags. We still don't know what specific tags do or why we want to give them things like IDs or classes, but that is coming very soon!

Additional Resources

Here are some references to learn more about the topics we discussed in this exploration.

- MDN's has detailed **introduction and tutorials on HTML** (https://developer.mozilla.org/en-US/docs/Web/HTML) .
- **All HTML elements** (https://developer.mozilla.org/en-US/docs/Web/HTML/Element) on MDN.
- **W3Schools** (https://www.w3schools.com/) also has great information on HTML (and other topics related to the web).
- A complete list of entities is available at the **HTML Living Standard site** (https://html.spec.whatwg.org/multipage/named-characters.html#named-character-references) .