10.2.1 Use HTML Elements

Robin has gotten all of her tools installed and tested Mongo to make sure it's ready for data, but before she can really start pulling it off of the web, she needs to be able to identify where the data is stored within the HTML code.

Every webpage is built using hypertext markup language, more commonly known as HTML. Some sites are more sophisticated than others, but they all have the same basic structure. Each element of a page, such as a title or a paragraph, is wrapped in a tag. Each tag is specific to the element it's holding, and there are many different types of tags.

Let's take a closer look at HTML tags.

Think of a webpage as a window into the internet. HTML is the glass, boards, and blinds on that window. Just like there are many sizes and shapes to windows, each webpage has been customized to present users with a view into a different topic. Consider a weather report delivered through a weather site. Think of a news source or social media platform. Each of these examples are all built using custom HTML. Our first step will be to explore that design so that we can write a script that knows what it's looking at when it interacts with a webpage.

Open VS Code and create a file named `index.html`. This file can be saved to your desktop because it's just for practice.

In this blank HTML file put an exclamation point on the first line and press Enter. This should autofill the editor to contain everything we need for a basic HTML page.

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <meta http-equiv="X-UA-Compatible" content="ie=edge">
 <title>Document</title>
</head>
<body>
</body>
</html>
```

In this code, each line of code is wrapped in a tag, such as `<title>`. Let's take a closer look at this tag to get a feel for HTML syntax.

HTML tags always begin with a left angle bracket (`<`), followed by the name of the tag (in our case, "title"). Once the name has been entered, the tag is then closed with a right-angle bracket (`>`). This is only the first half of the completed tag, or the opening tag. We'll also need to add the closing tag.

A closing HTML tag is very similar to the opening tag, but the only difference is a single character, the forward slash inside the left angle bracket: `</title>`. Now that there are both opening and closing HTML title tags, you can add the title of the document.

For example, if you wanted your webpage title to be "Math Is Fun!" then the entire line of HTML code would look like this: `<title>Math Is Fun!</title>`.

NOTE

> Sometimes you'll see tags like this: `<title />`. This isn't a new-fangled way of presenting HTML code, it's just a way of summarizing the tags and their contents. This is more commonly seen in written descriptors of HTML and not in live HTML code.
>
> You'll see different tags referred to in the same manner as you learn how to create and customize webpages.

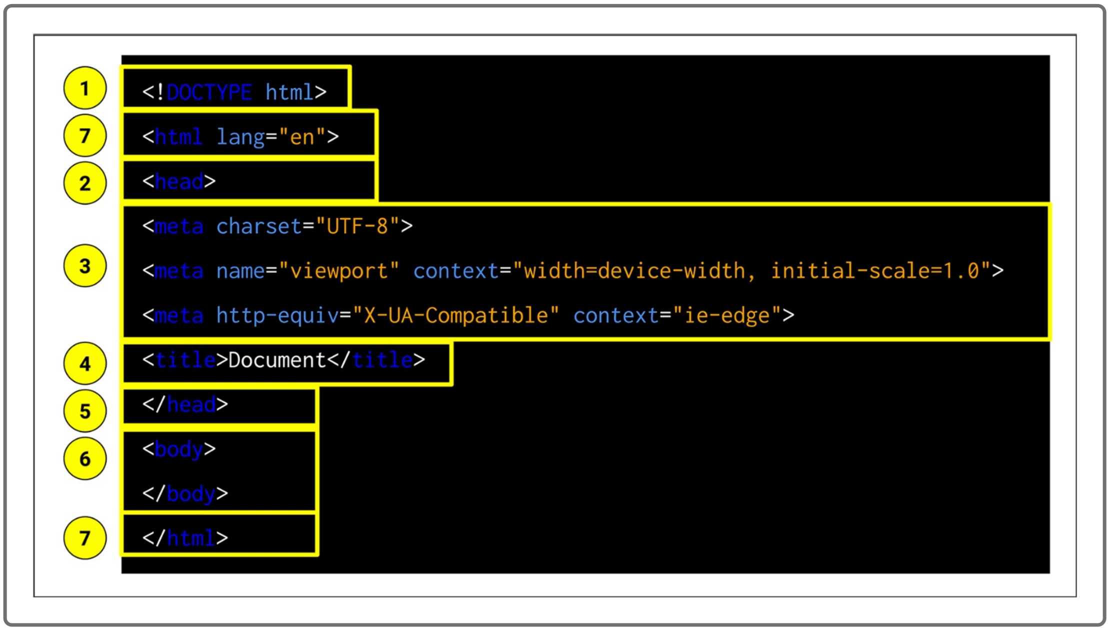
REWIND

HTML is a coding language used for creating webpages. It's built using specific tags and arranging them in a nested order, a bit like building blocks. For example, if we wanted a header and a paragraph in the same section of a webpage, we would nest `<h1 />` and `<p />` tags inside a `<div />` tag, with the `<div />` tag acting as a box to hold the other pieces.

```
<div>
   <h1>Hello, world!</h1>
   <p>This is a great beginning.</p>
</div>
```

Most elements have opening and closing tags, which are identical except for the forward slash that begins the closing tag. The closing tags represent the end of that HTML element.

These tags are what define each element of this webpage. We can open this page right now, but it will be blank because we haven't added anything to it yet. Let's take a closer look at how these different elements work together.



```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" context="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" context="ie-edge">
<title>Document</title>
</head>
<body>
</body>
</html>
```

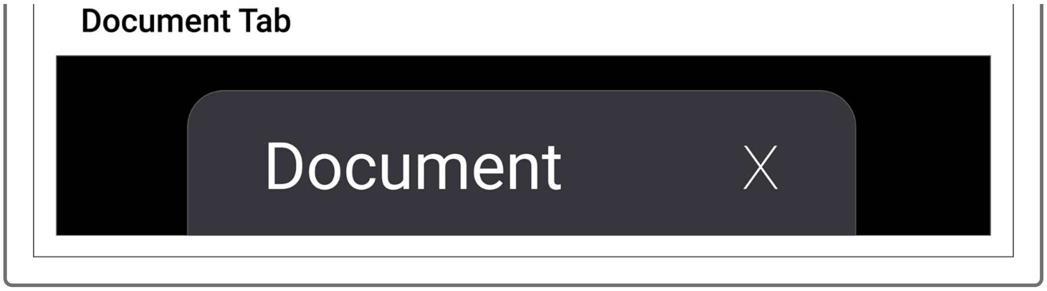
Let's define each HTML tag shown in the graphic:

1. `<!DOCTYPE html>` is a declaration, not a tag. It tells web browsers in which HTML version the document is written. This should always be the first line in an HTML document.

2. `<head>` is the opening tag that serves as a container for the setup elements. Jupyter Notebook imports occur in the top cell whereas Python imports occur at the top of the code. HTML imports (e.g., a stylesheet or a library) will be within the `<head>`.

3. `<meta>` is short for "metadata" and tells the web browser basic information, such as page width.

4. `<title>` and `</title>` are the opening and closing tags that serve as a container for the page title displayed on the tab at the top of your web browser. In the example above, the title is "Document" and would appear like so in the browser:



5. `</head>` is the closing tag for the `<head>` tag, much like the end of a code block in Python.

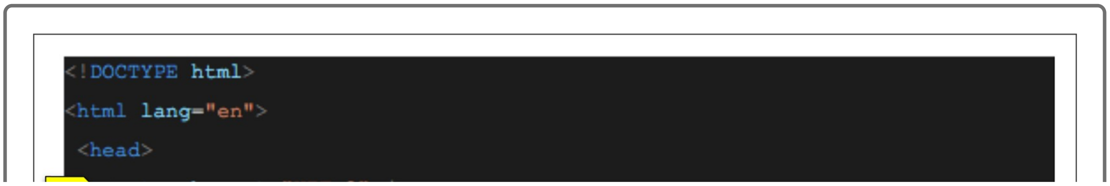
6. `<body>` and `</body>` are opening and closing tags. They also serve as a container, but for data we can see (navigation menus, lists, and paragraphs).

7. `<html lang=”en”>` and `</html>` are opening and closing tags that serve as a container for all elements within an HTML page.

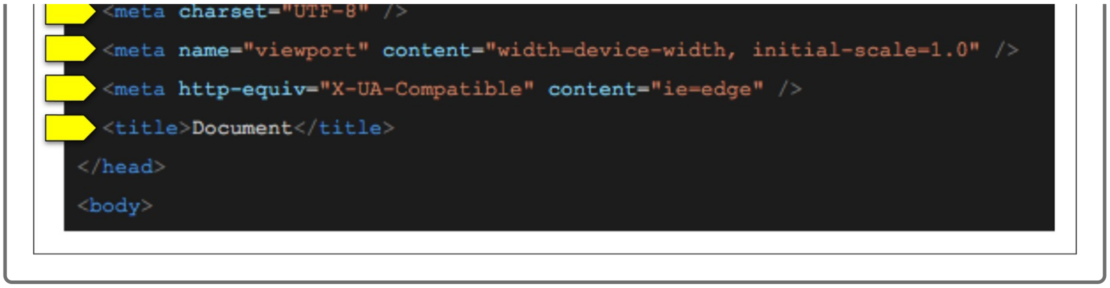
IMPORTANT

> Nesting is when HTML elements are contained within other elements. Picture a set of nesting dolls with each nested in proper order, by design, into the largest doll. It is the same for HTML tags—they must be in the correct order to not break the design of the webpage.

An easy way to keep the tags in visual order is by using indentation. Containers nested within other containers are indented by two to four spaces. This helps to keep our code clean and easy to understand.



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



```
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta http-equiv="X-UA-Compatible" content="ie=edge" />
  <title>Document</title>
</head>
<body>
```

Let's take another look at this webpage, only with a few more elements added to it:

```
<!DOCTYPE html>
<html lang="en">
 <head>
   <meta charset="UTF-8" />
   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
   <meta http-equiv="X-UA-Compatible" content="ie=edge" />
   <title>Document</title>
 </head>
 <body>
   <h1>Hello, world!</h1>
   <p>
     Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin aliquet
     iaculis lorem non sollicitudin. Fusce elementum ac elit finibus auctor.
     Curabitur orci sem, accumsan a diam sit amet, efficitur tristique velit
   </p>
   <ul>
     <li>First list item</li>
     <li>Second list item</li>
     <li>Third list item</li>
   </ul>
 </body>
</html>
```

There are several more tags within the `<body />` container. Add this new code to your `index.html` file and save it. Then, open the file by navigating to it and double-clicking it. Now you have a simple static webpage open in your browser, built from scratch. It's not super exciting yet, but that's okay. It's the innards of the page we're focusing on right now.

Let's review the new tags:

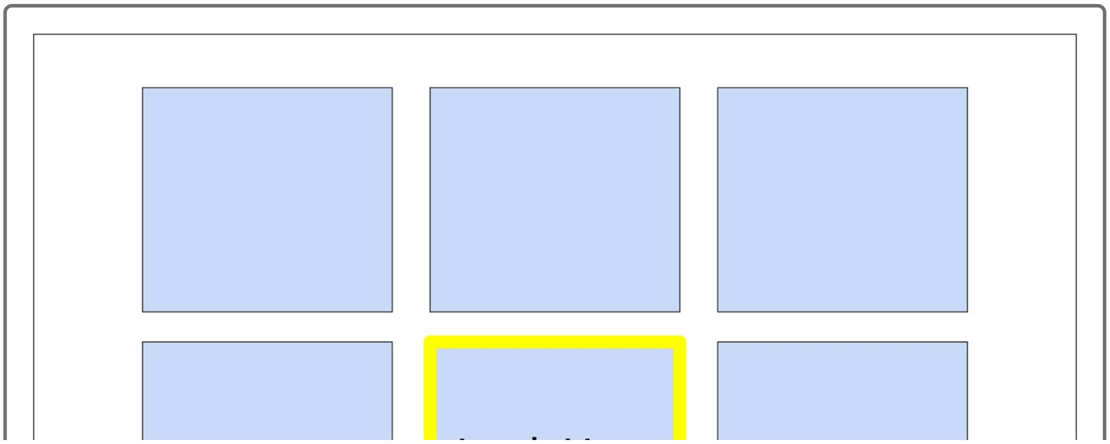
- `<h1 />` is a first-level header. The text in this tag will be displayed bigger and bolder than the rest of the page's text. There are many different headers available to use, from `h1` to `h6`, with `h1` returning the largest text.
- `<p />` is a paragraph tag, currently holding **lorem ipsum** sentences. (lorem ipsum is dummy text used to stage websites). More can be read about it on the [Lorem Ipsum reference website (https://www.lipsum.com/)](https://www.lipsum.com/) .
- `<ul />` is an **unordered list**.
- `<li />` is a **list item**.

Loading. Please Wait...

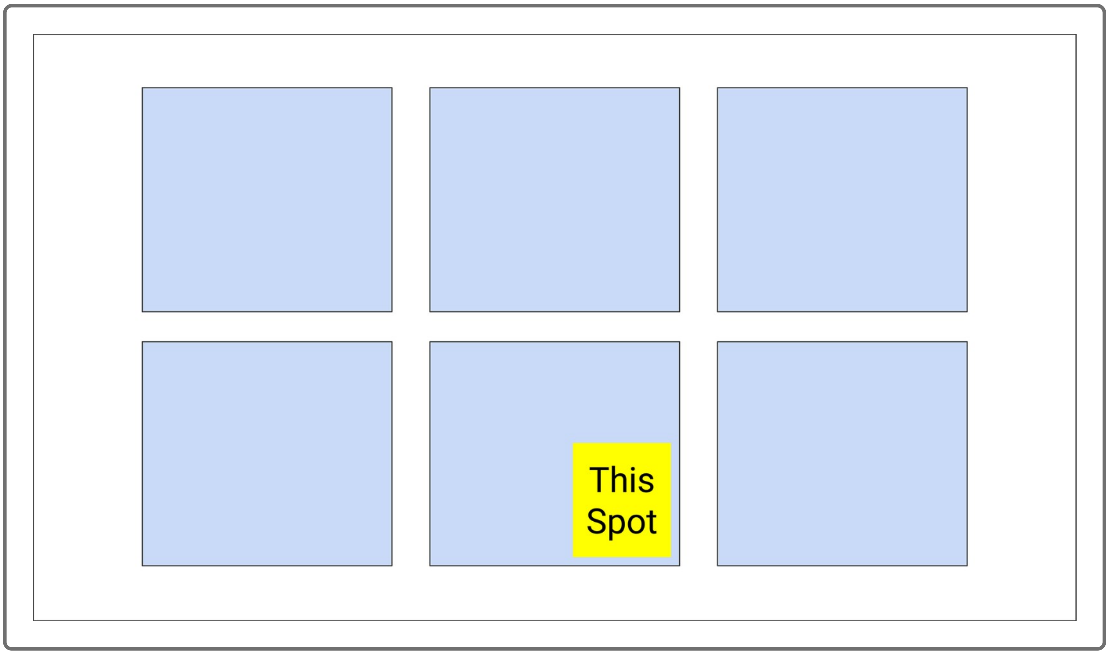
This is only a small taste of how many tags exist out there. Remember, these tags are all part of website customization. Without the variety available to use, websites would look plain and uninspired. The sites that Robin intends to scrape data from are far more sophisticated, using many more combinations of tags than what we've discussed here. Understanding the basic layout and how nesting and containers work is an important part of successful web scraping.

We know that when we scrape data from the web, we're simply pulling specific data from websites we've chosen. How do we specify the data? Let's say we want the latest news article from a Mars website. Before we can program our script to pull that data, we have to tell it where to look. Basically, our script would say, "look in this `<div />` tag, then look inside that for a `<p />` tag."

For example, if the webpage was a window, we would use our script to direct it to a certain pane.



Once it found that pane, we can also tell it to look even closer, such as at the bottom-center pane.



That's a simple way of putting it, but we'll dive more deeply into how web scraping works soon. Visit W3Schools' developer site for an extensive list of **HTML tags** (https://www.w3schools.com/tags/tag_comment.asp) .