20.1. Background

When programmers make web pages, they want their pages to be beautiful, interactive, and fun. Programmers may use JavaScript to make their pages interactive, but JavaScript does not do much to define the structure and appearance of a web page. The next two chapters cover HTML and CSS, which are the two most common languages for structuring content and making it beautiful.

Before jumping in to learn HTML and CSS, we need to understand how web pages appear on screens. The process involves the browser and the server that hosts the code. You are probably very familiar with browsers as the tool that gives us access to the internet. However, programmers think of browsers a little differently. For them, the browser is what translates the code into a web page.

When you visit a web page in a browser, three main steps happen:

1. The browser sends a **request** to the server for the web page.
2. The server **responds** with the code that makes up the web page.
3. The browser takes the code and renders it to present the web page that the code creates.

When the browser renders the page, HTML outlines the structure of the page's content.

Note

In later chapters, request and response between browsers and servers will be covered in greater detail.

20.1.1. What is HTML?

Indicators of how HTML works are in its name. HTML is short for Hypertext Markup Language.

Hypertext is text that includes references to other text known as hyperlinks.

With coding languages, there is a family of languages called **markup languages**. Markup languages annotate the text of a document and define the structure. HTML is the markup language that defines the structure of hypertext.

HTML's two main components, elements and tags, are key to defining the structure of content.

20.1.2. HTML Elements

When a programmer creates a web page, they break the content down by type. They may outline a structure for the page on paper first, highlighting what each item is. With HTML, a programmer can add a lot of different types of content to a page. In this chapter, the focus is on headings, paragraphs, images, and more.

An **element** is a segment of an HTML page. Elements are oftentimes broken down by content type.

20.1.3. HTML tags

An HTML **tag** is the syntax that the computer processes to determine the type and content of an HTML element.

Tags surround the content within the element, so in all cases, programmers need to have opening and closing tags.

Each tag has the following structural elements:

1. `<` to start a tag and `>` to close it.
2. The type of element it is.
3. Optional additional specification about the element's appearance.
4. Closing tags include the same information as the opening tag with a `/` after the `<` bracket.

Here is an example of a line of HTML:

```
<element type>content</element type>
```

20.1.4. HTML Writing Style

Programmers write HTML different ways with different style guides and philosophies. **Semantic HTML** is not about the appearance of the web page, but about the specific meaning of the elements. Semantic HTML helps programmers communicate through code and may be easier to pick up at first. Programmers can make a paragraph larger than a heading. But by looking at the HTML, another programmer can understand which is the paragraph and which is the heading. Another benefit to semantic HTML is that it is easier for beginning programmers to visualize the end results. Some examples of semantic HTML tags are: `<p>`, `<h1>`, `<h2>` , and `<div>`.

Reminder

Making code work is important and so is making it easier for other programmers to read. Not every piece of code a programmer reads is something they wrote.

20.1.5. Check Your Understanding

Question

What does HTML stand for?

a. Happy Tickles Make Laughter
b. Hypertext Markup Language
c. Hypertext Mockup Language
d. Hyperlink Markup Layout

20.2. HTML Structure

Programmers should follow certain rules about how to structure an HTML file. The rules about how to structure an HTML file and the tags used to lay out this structure are vital to the browser being able to render the page.

20.2.1. Structure Rules

When it comes to laying out the overarching structure of an HTML file, a programmer should follow 5 rules:

1. Every HTML file needs a `DOCTYPE` tag, specifying the HTML version used. When using the current version of HTML, the `DOCTYPE` tag is simple to remember as it is: `<!DOCTYPE html>`. This is one of few tags that does not require a closing tag.
2. The `<html>` tag denotes the beginning and end of the HTML the programmer has written.
3. The `<head>` tag contains data about the web page.
4. The `<body>` tag contains everything that appears on the page of the document.

5. The `<title>` tag goes in the `<head>` of the document and browsers require it. It gives the title of the webpage that appears in the tab.

Here is an example of the structure of an HTML page based off of these rules:

```
<!DOCTYPE html>
<html>
   <head>
      <title>My Web Page</title>
      content
   </head>
   <body>
      content
   </body>
</html>
```

20.2.2. Document Head

So other than the title, what goes in the head of an HTML file? The head includes links to other files and other data about the document. Browsers do not display the content in the head.

Note

The head can also include some styling to make the page beautiful. How to do that is covered in the next chapter on CSS.

20.2.3. Document Body

After the programmer has written the head of the document, it is time to move on to the body of the document. The body of the document contains the content that appears on the web page. Within the `body` tags, programmers add images, text, and even code samples with different HTML tags. Content outside of the body will not appear on the page.

To make HTML more readable to other programmers, programmers write comments in HTML. When adding a comment, the programmer uses `<!--` to indicate the start and `-->` to end the comment, like so:

```
<body>
   <!-- This is an important comment -->
</body>
```

Note

Spacing and tabs helps many programmers read through theirs and their colleagues' code. Be aware that doing so in HTML can effect how the browser renders the page in rare instances.

20.2.4. Check Your Understanding

Question

Which HTML tag does not require a closing tag?

a. `title`
b. `body`

c. `head`
d. `DOCTYPE`

20.3. HTML Tags

Time to dive into learning about all the different tags for creating content! This page contains a helpful table of tags to know for beginning programmers to bookmark. This is by no means an exhaustive list of all HTML tags, but it is a good place to start.

20.3.1. Tags to Know

Tag Name	Code	Definition
Bold	`<b>`	When surrounding text, makes that text bold.
Emphasis	`<em>`	When surrounding text, makes that text italic.
Hyperlink	`<a>`	Creates hyperlinks.
Image	`<img>`	Denotes images.
Break	` `	A single line break.
Paragraph	`<p>`	Creates a paragraph in text.
Section	`<span>`	Makes a section in text.
Division	`<div>`	Defines an area of the page.
Form	`<form>`	Creates a form for user input.
Unordered List	`<ul>`	Creates an unordered list.
Ordered List	`<ol>`	Creates an ordered list.
List element	`<li>`	Denotes an element of the list. This tag is used for both ordered and unordered lists.

Tag Name	Code	Definition
Table	`<table>`	Creates a table on the page.
Heading Level One	`<h1>`	Creates a heading in the text.

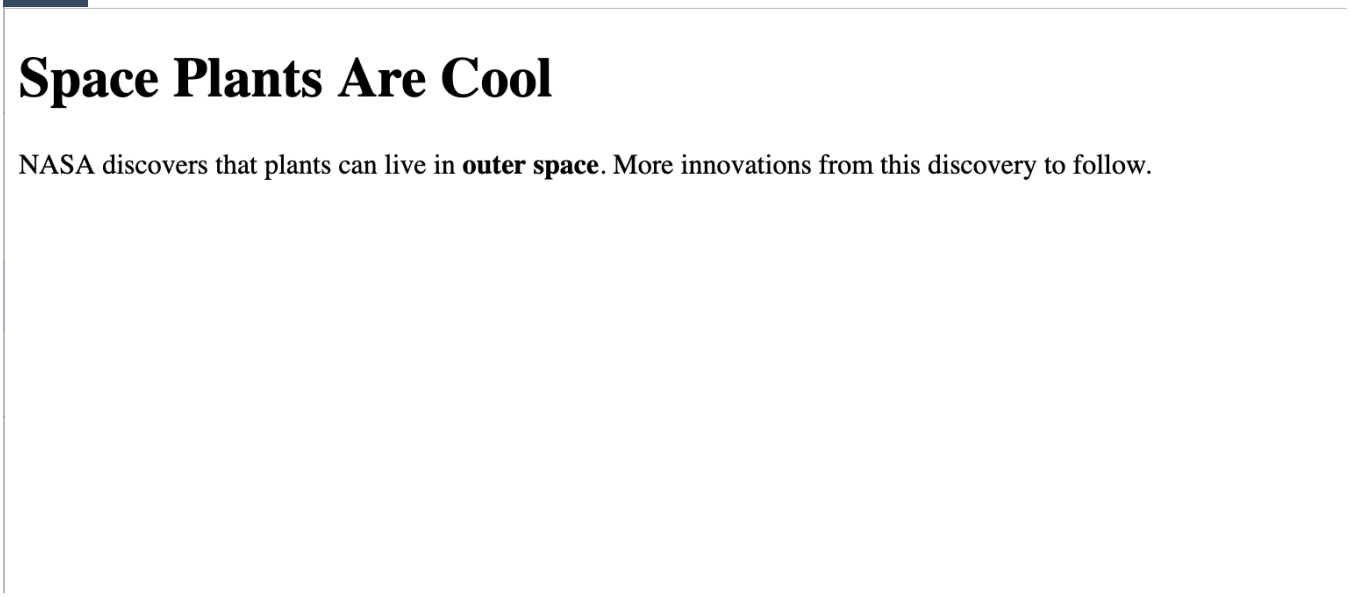
Note

There are multiple headings in HTML going from `h1` to `h6`. The headings get progressively smaller. A good rule of thumb is to have only one `h1` in a web page and do not skip a level. Headings can be resized so there is no need to do so.

20.3.2. Tag Example

Here is an example of a basic web page utilizing some of the tags above with the HTML used to make the site.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Plant-Loving Astronauts</title>
  </head>
  <body>
    <h1>Space Plants Are Cool</h1>
    <p>NASA discovers that plants can live in <b>outer space</b>. More innovations from this discovery to follow.</p>
    <!-- add images from NASA of these space plants -->
  </body>
</html>
```



20.3.3. Attributes

Programmers can add extra information beyond element type to HTML tags. Programmers add **attributes** to HTML tags for further specification about the element's appearance. Examples of attributes include the alignment of the element or alternate text to an image.

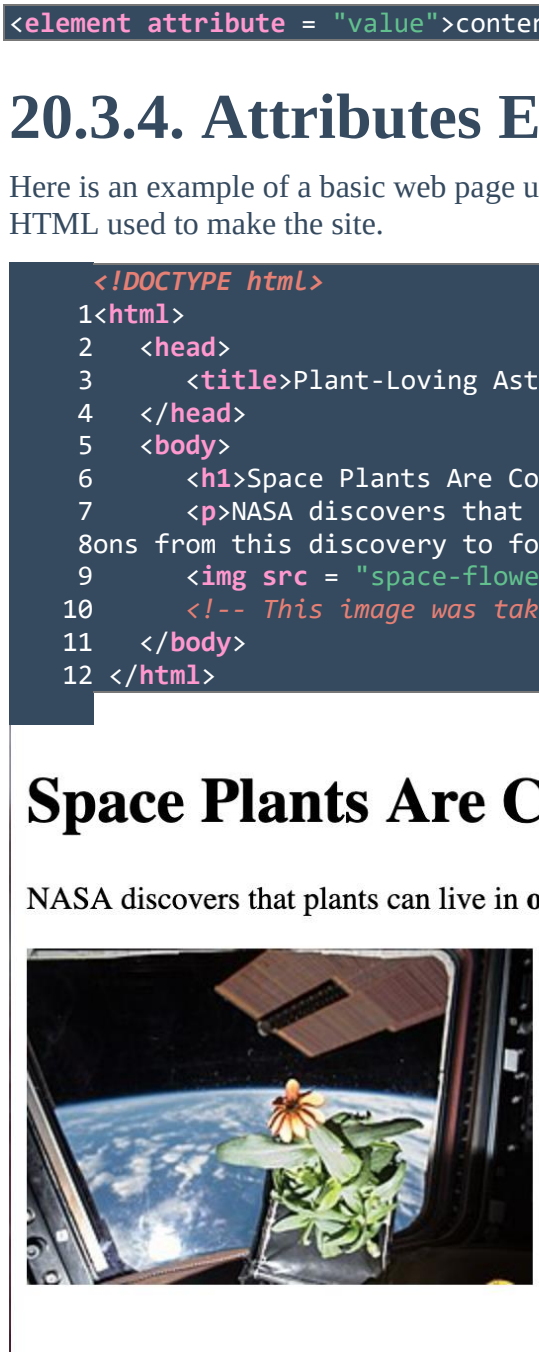
Programmers add attributes before the closing bracket in the opening tag, like so:

```
<element attribute = "value">content</element>
```

20.3.4. Attributes Example

Here is an example of a basic web page utilizing some of the tags above and appropriate attributes with the HTML used to make the site.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Plant-Loving Astronauts</title>
  </head>
  <body>
    <h1>Space Plants Are Cool</h1>
    <p>NASA discovers that plants can live in <b>outer space</b>. More innovations from this discovery to follow.</p>
    <img src = "space-flower.jpg" alt = "Flower floating in space.">
    <!-- This image was taken by NASA and is in the Public Domain -->
  </body>
</html>
```



The `<img>` tag has two attributes that you will see a lot. `src` gives the location of the image that is being used and `alt` gives alternate text for screen reader users. For that reason, `alt` should be a concise description of what is going on in the image.

20.3.5. Check Your Understanding

Question

Which tag is used to make text italicized?

a. `b`
b. `i`
c. `em`
d. `br`

20.4. Exercises: HTML

Complete the HTML file for this simple webpage. Add lines to `index.html` that do the following.

1. Add a `h1` to the page that says “Why I Love Web Development”
2. Add an ordered list to the page with 3 reasons why you love web development.
3. Add a link to this page below your list.
4. Add a paragraph about the website you want to make with your web development superpowers!

This code block gives you a rough outline for how it might look.

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <!-- h1 goes here --->
    <!-- ol goes here --->
    <!-- a goes here --->
    <!-- p goes here --->
  </body>
</html>
```

repl.it

Note

repl.it has other HTML inside of the `index.html` file you will be editing. You should not be deleting any code only, only adding code to the file!

20.5. Studio: Making Headlines

20.5.1. Getting Ready: Developer Tools

As you've learned, debugging is an essential part of coding. When it comes to debugging web pages, browser developer tools are indispensable.

This studio requires you to use Firefox's developer tools. In particular, you should be able to:

- Open Firebox's dev tools
- Inspect an HTML element
- Modify an element's HTML
- Explain the difference between the content displayed when using *View Source* and what you see in the *Inspector* tab

Note

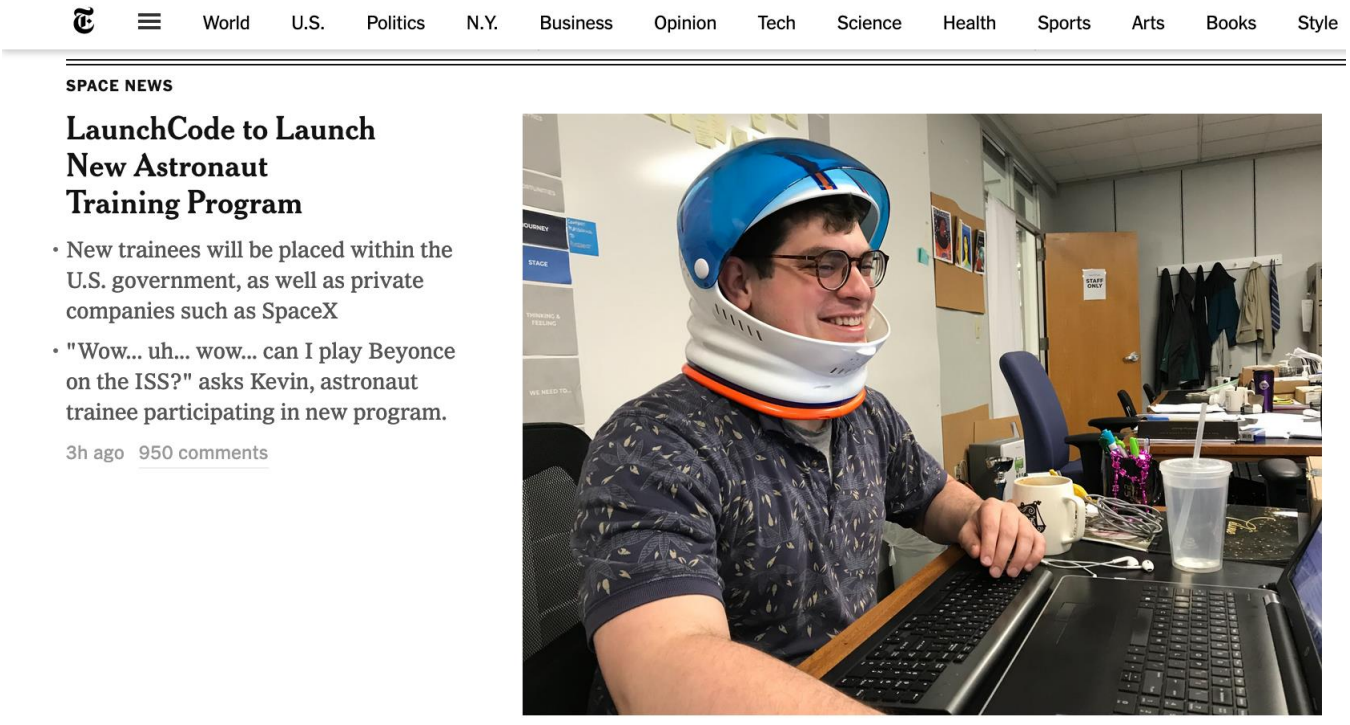
The full documentation for Firefox's developer tools covers these items, and much more.

20.5.2. Studio

Pick a news site (The New York Times, for example), and use your browser's developer tools to modify one of the main articles to use a picture and text of your choosing.

Have fun with this, but be respectful of others and avoid overtly critical political or social commentary.

You might do something like this:



A Sample Fake Article

20.5.2.1. Image URLs

When linking to an image, pay attention to the protocol of both the site you are modifying and of the image you are including. The protocol will be either `http` or `https`.

If the site loads over `https` and your image uses `http` then the image may not load properly due to browser security restrictions. You should try to add `s` to the image protocol, and if that doesn't work, look for another image.

If you want to use an image of your own that is not already available via the internet, here's how:

- Upload the photo to a Dropbox account
- View the photo on Dropbox and select *Share*, then *Get link*, then *Go to link*
- You should now be viewing the image on the Dropbox site. If the URL contains `?dl=0`, remove it. Add `?raw=1` to the end of the URL in the location bar of your browser and hit *Enter*. The URL should look something like this:

```
https://www.dropbox.com/sc/qc3htnhv7fb3i2x/AAC5OzECOyBynstMDWawCZhxa?raw=1
```

- Copy the URL of the loaded image. You can use this URL within any HTML.

20.5.3. Resources

- Using Firefox's Page Inspector
- Firefox DevTools Documentation