

# HTML Document Structure

by Sophia



## WHAT'S COVERED

In this lesson, you will learn about the basic “core” structure of code required for every webpage. You will also learn about what each section of the core structure is responsible for and what HTML elements generally belong within. Many of the common tags used throughout webpages will also be introduced. These tags will form the foundation for you to begin building content into a webpage’s <body> section.

Specifically, this lesson will cover the following:

### 1. Core Structure

#### 1a. head

#### 1b. body

### 2. Common Tags

## 1. Core Structure

All HTML webpage documents use the same core structure of HTML elements. This structure is dictated by the HTML standard and includes the following four minimum elements.

### 🔗 EXAMPLE

```
<!DOCTYPE>
<html>
<head>
<body>
```

These elements are structured together in the same pattern for each webpage. The following is an example of a completely empty but fully valid HTML5 webpage structure. It is the very beginning of the creation of a webpage.

### 🔗 EXAMPLE

```
<!DOCTYPE html>
<html>
  <head></head>
  <body></body>
</html>
```

There are two elements at the root of the document (the outermost elements): the **<!DOCTYPE>** and the **<html>** elements. When a browser is going to render a page, the first thing it needs to know is the type of document it is rendering. Furthermore,

since browsers read an HTML document from top to bottom, the very first line of code in an HTML document is `<!DOCTYPE>`, a self-closing tag that specifies what kind of document it is and what version. This statement allows the web browser (i.e., Chrome, Firefox, etc.) to know what version the HTML document is written in so it can render the page appropriately.

#### ↪ EXAMPLE

The latest version of the HTML version 5 doctype declaration is as follows:

```
<!DOCTYPE html>
```

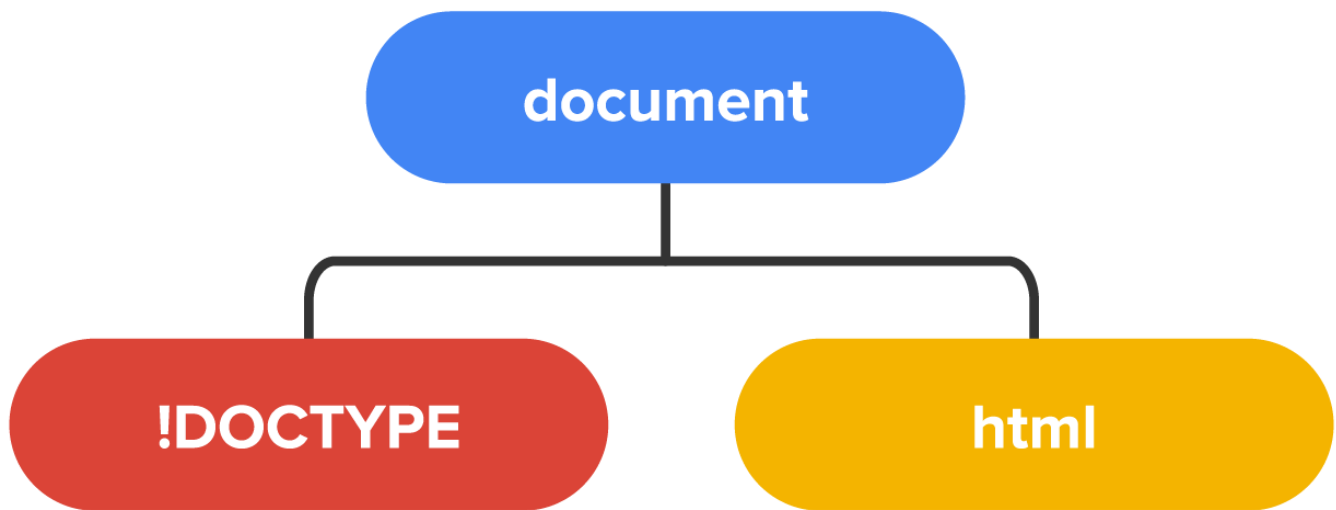
Notice that the convention is to capitalize doctype, while all other tag names are lowercase.

In the previous version of HTML, the doctype statement was much longer and more complicated:

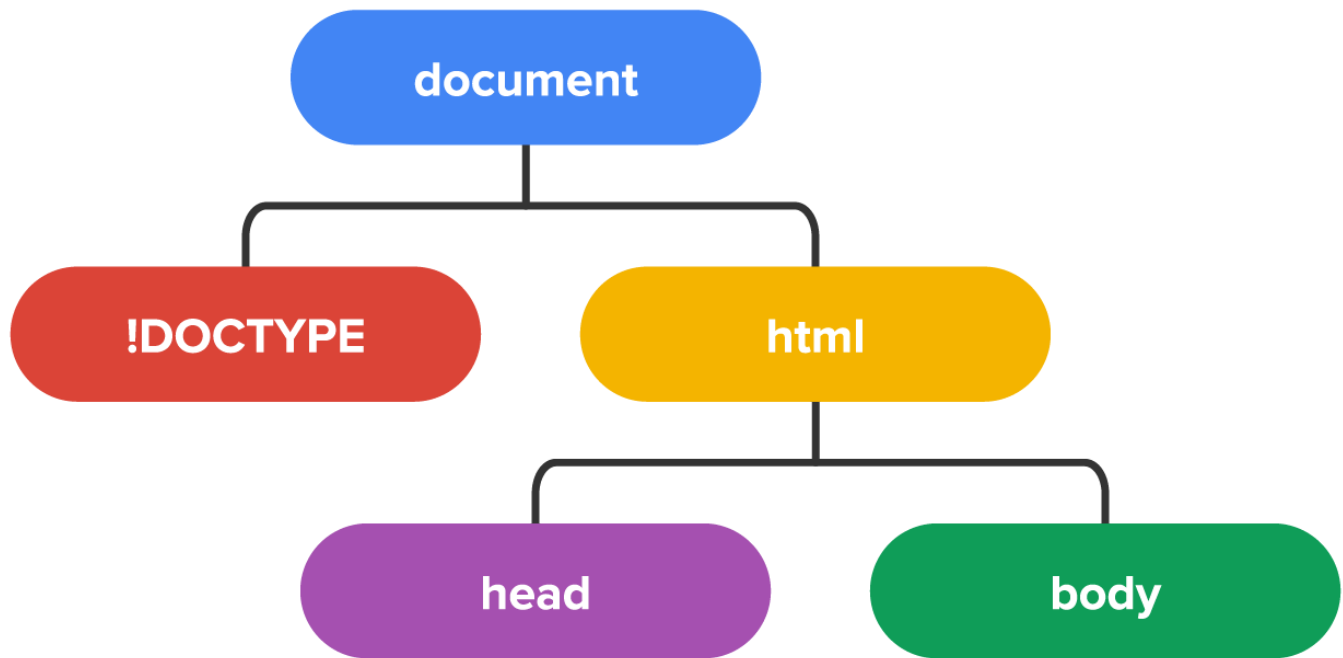
#### ↪ EXAMPLE

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

The `<html>` element contains all of the content and settings for the webpage.



Next down the line, you will see that there are two child elements within the `<html>` element: `<head>` and `<body>`.



The `<head>` element is meant to contain all of the configuration information for the page as well as link to external resources. The head is also used to contain an internal stylesheet for the page, whenever needed. With the exception of any dynamic content generated by JavaScript code located in the head, nothing within the `<head>` will be visible on the screen.

The `<body>` element will contain all of the visible content that will be displayed in the web browser. Any text written within the `<body>` tags will be displayed on the webpage screen as plain regular text.

It is considered good practice to manually write the HTML page structure, including the symbols and indentation that you see above.



**Directions:** Take a sheet of paper and a pencil or pen, and write the following HTML code.

#### ↪ EXAMPLE

```
<!DOCTYPE html>
<html>
  <head></head>
  <body></body>
</html>
```



Writing out code by hand will help you become familiar with the code structure, symbols and syntax, as well as the indentation of the code. Furthermore, the more you practice writing and manually typing HTML code, the better. The common errors of all new coders and programmers are most often typos in the code. Some common mistakes made are listed below:

- Misspelling tag names, attributes, and values
- Forgetting to close tags with the `>` symbol
- Forgetting to close the attribute value's quotes

A good practice for when you run into coding issues and your page content is not rendering correctly or at all (or worse, you see raw HTML code on the webpage) is to examine your code, line by line, word by word, and character by character. Additionally, use a working example to help guide you on how your code should look.

## TERMS TO KNOW

### **DOCTYPE Declaration <!DOCTYPE>**

A special tag used to indicate the type and format of the digital document.

### **Head Tag <head>**

An HTML tag used to define the head portion of a webpage, which is used to configure the page and link to resources in external files.

### **Body Tag <body>**

An HTML tag used to define the body of content of a digital document, which is used to create all of the visible content that will be displayed on the page.

## **1a. head**

The <head> section of a webpage is used for the setup and configuration of the page. The <head> is also used to import external resources such as stylesheets and JavaScript script files. The common tags used within the head to accomplish many of these tasks, and more, include the following:

- <title>
- <link>
- <style>
- <meta>
- <script>

The <title> tag is used to set the title of the document. The title does not appear in the webpage content; rather, the title is what is shown in the title bar or tab of the browser window itself. If you currently have multiple tabs open within your browser, the title you see written in the tabs is the title created by including the tag. Commonly, the title includes the name or title of the organization, a dash or some other delimiter, and then the title of the page itself.

### EXAMPLE

```
<head>
  <title>Good Harvest Bakery - Home</title>
</head>
```

<link> is a self-closing tag used to bring in resources located in other files. Commonly, the <link> tag is used to link to an external stylesheet like so:

### EXAMPLE

```
<head>
  <title>Good Harvest Bakery - Home</title>
  <link rel="stylesheet" href="styles.css">
</head>
```

The `<link>` tag can also be used to set the site's icon that appears just before the title on browser tabs. This icon is often referred to as the "favicon" as it also appears next to each bookmark in your browser's bookmark, or favorites, list.

#### ⇨ EXAMPLE

```
<head>
  <title>Good Harvest Bakery - Home</title>
  <link rel="stylesheet" href="styles.css">
  <link rel="favicon" href="favicon.ico">
</head>
```

The `<style>` tag, as you may recall, is used to create an internal stylesheet specific to the page.

The `<meta>` tag is used to provide a variety of additional information about the specific webpage. For example, meta is used to specify the **character set** of the HTML file, the page description, keywords, the author, and viewport settings. The keywords, author, and description are used by search engines to categorize and index your page's content. However, the viewport settings and character set have more impact on the page itself. Character encoding ensures that the process used by your computer to encode the text characters is compatible with other systems.



#### KEY CONCEPT

Character set encoding is a specification related to how the actual file's contents are encoded by the computer. There are many different character sets, including ASCII, UTF-8, and Windows-1252. The word "Hello" encoded in ASCII or Windows-1252 may result in incompatibilities when rendered on a machine in Japan running in Japanese. This may cause unrecognized or unencoded symbols to appear as random characters on the page, often as empty rectangles. UTF-8 is one of the most widely compatible character sets. Most webpage files are encoded using UTF-8, and as long as the character set is specified by a meta tag, there will be no compatibility issues with text.

The `<script>` tag is used to embed JavaScript into the page. When the `<script>` tag is added to the head, the JavaScript code contained within is executed as soon as the webpage is loaded into the browser. Oftentimes, **JavaScript functions** (named set of JavaScript instructions) are placed in the head within the `<script>` tags for use by the body. These JavaScript functions do not execute until something in the body of the page calls them.



#### TERMS TO KNOW

##### Title Tag `<title>`

An HTML tag that goes in the `<head>` section of a webpage that sets the title value of the web browser tab.

##### Link Tag `<link>`

An HTML tag used to link to either an external stylesheet or an icon image file for the webpage.

##### Style Tag `<style>`

An HTML tag used to embed a section of CSS style rules within the `<head>` section of a webpage that only affects the page in which it is contained.

##### Character Set

A system of numbers used to encode text-based characters for storage and retrieval of text data within a computer file. Also called *character encoding*.

##### Script Tag `<script>`

An HTML tag used to embed JavaScript code directly within an HTML document.

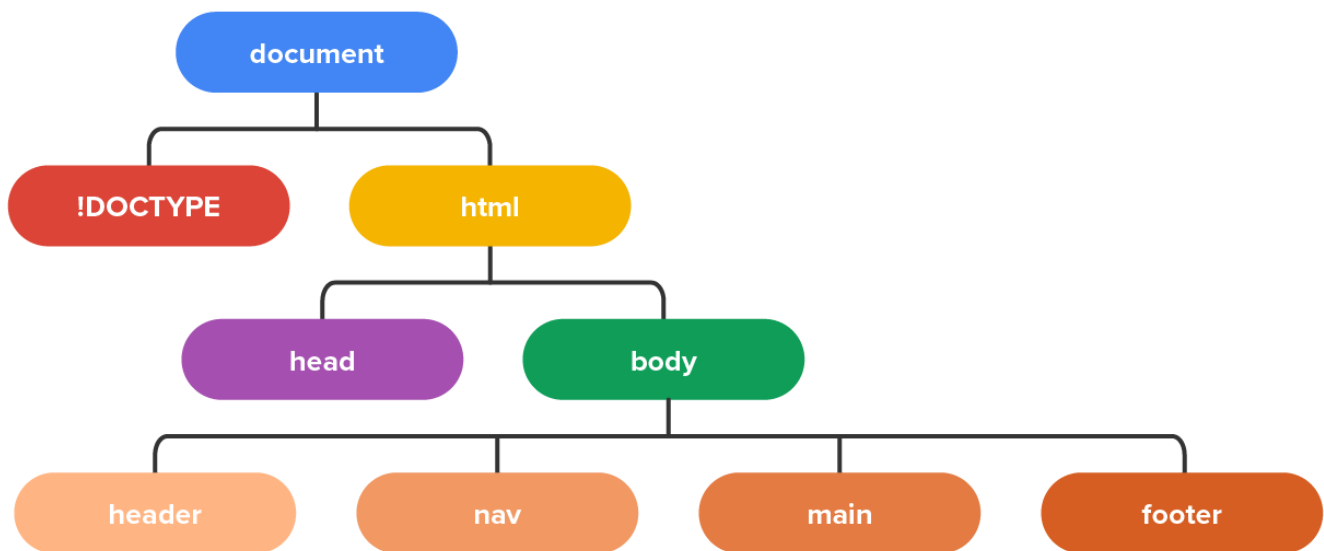
## JavaScript Functions

A named set of instructions that can accept data as well as return an output value, written in JavaScript, and can be called as many times as possible, whenever needed.

### 1b. body

The body section node is where all visible content is placed. It includes the semantic tags that give structure and meaning to the different sections of the visible content and other tags and their attributes to provide layout and behavior control.

Think back to the DOM structure; the next level of child nodes of the body will be the semantic tags. Remember that semantic tags are tags meant to provide meaning and purpose to the sections of the webpage and the sections of associated code.



Remember that the semantic tags are not limited to just header, nav, main, and footer, and you will include any of the semantic tags as needed by the webpage.

Within each of the semantic tags, you add content with additional structural or formatting tags. For example, in the <header>, you will likely include the page's top heading or title within an <h1> element, in addition to a logo and possibly a call to action (CTA). The <nav> section will include the hyperlinks to navigate the website. The <main> section will contain the main content for the specific page. The <footer> will be used to provide a typical footer containing copyright information, simplified navigation hyperlinks, social media links, etc.

Additional semantic sections such as <section>, <article>, or <asides> can be added to provide sidebars or main sections with multiple subsections.

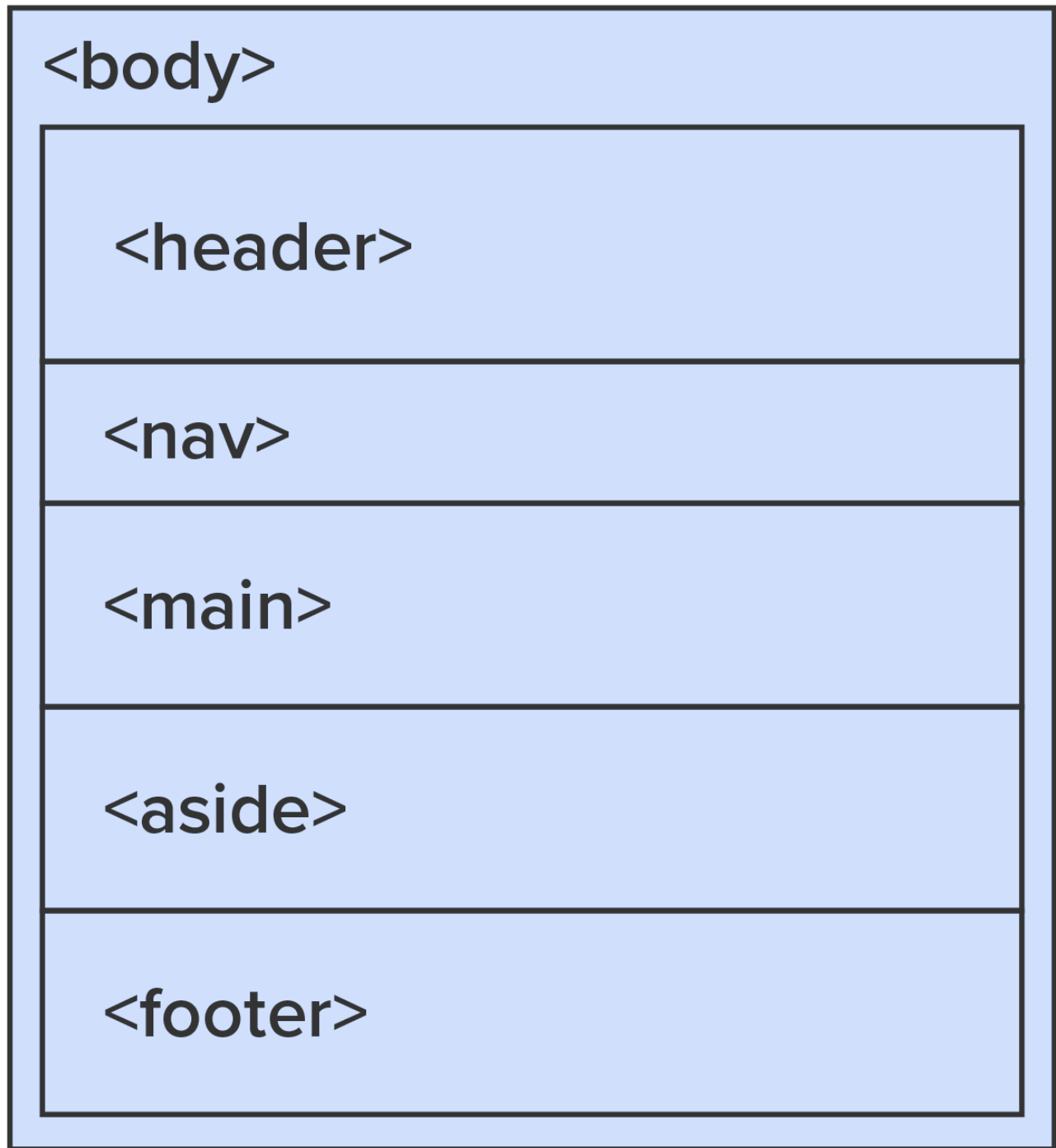
Other elements such as tables, ordered and unordered lists, images, and script tags are also contained within the semantic section of the body. It is worth noting that the <script> element can also be placed within the body and is used to hold JavaScript code, just like the <script> element in the <head>. However, the code within a <script> element located within the body is executed in place at the time the browser reads and renders the <script>.

It is good to note at this point that these semantic sections will simply take up the full width of the screenspace and simply stack on top of each other.

🔗 **EXAMPLE** Here is the HTML <body> section code:

```
<body>
  <header></header>
  <nav></nav>
  <main></main>
  <aside></aside>
  <footer></footer>
</body>
```

↪ **EXAMPLE** This is how the semantic sections will appear on the page:

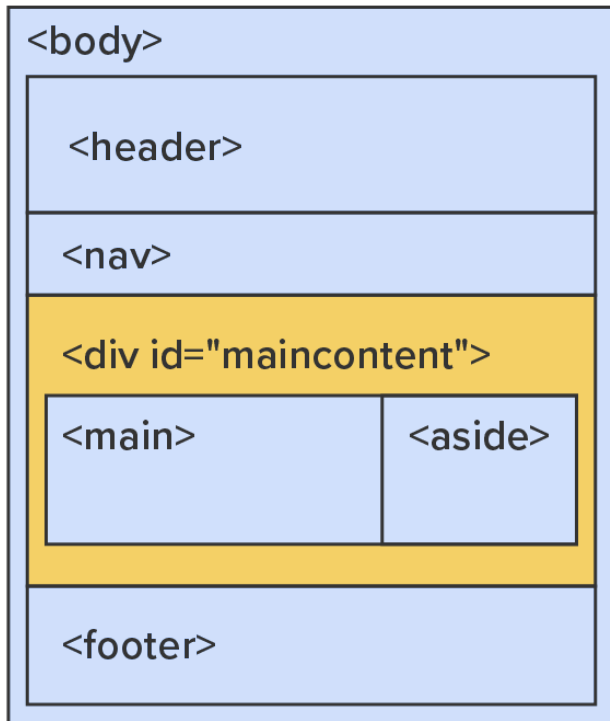


This is where CSS will come in. You will learn how to use CSS Flexbox in the next tutorial to control the layout of the semantic section and more.

#### 🔗 EXAMPLE

To locate the `<aside>` to the right or left of the `<main>` section, you will create a generic flex container around the semantic tags, set their width, and set `flex-direction` to `row` to make your page layout more like this:





### HTML source code:

```
<body>
  <header></header>
  <nav></nav>
  <div id="maincontent">
    <main></main>
    <aside></aside>
  </div>
  <footer></footer>
</body>
```

### HTML source code:

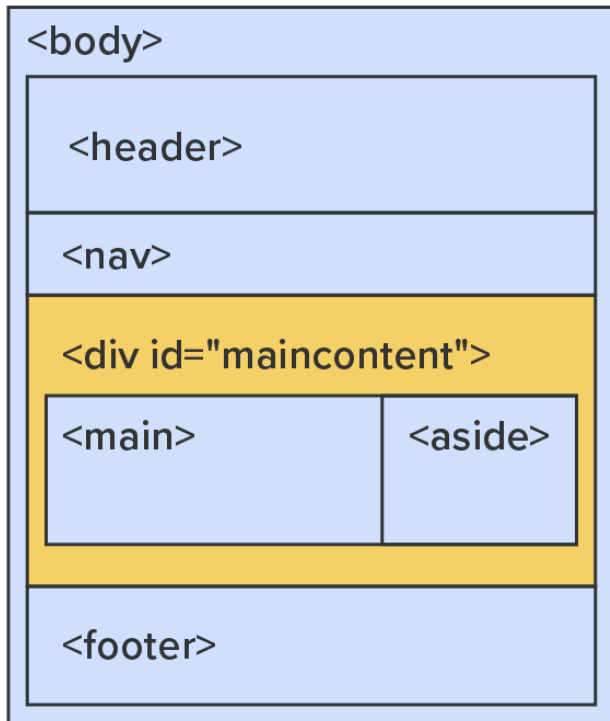
```
#maincontent {
  display: flex;
  flex-direction: row;
}
main { width: 70%;}
aside { width: 30%;}
```

Now that you have learned about the semantic tags and their place in the document, it's time to try it yourself.



#### Directions:

1. Return to your StackBlitz.com account, and create a new project from the starter templates, make sure to select "Static HTML/JS/CSS"
2. Remove all the content between the `<html>` tags. You will be building the following structure using semantic tags:



## HTML source code:

```

<body>
  <header></header>
  <nav></nav>
  <div id="maincontent">
    <main></main>
    <aside></aside>
  </div>
  <footer></footer>
</body>

```

3. Add the `<head>` and `<body>` sections within the `<html>` tags. Remember, these elements are not nested within each other; they are siblings and both are nested within the `<html>` tags.

This is the expected input and output (output should be blank):

```

1 <!DOCTYPE html>
2 <html>
3
4 <head></head>
5
6 <body></body>
7
8 </html>
9

```

4. Next, in between the `<head>` and `</head>` tags, add the following tags:

- The `<title>` element and give it some text such as "MyFirstPage - Home"
- The `<meta>` self-closing tag and give it the attribute "charset" with a value of "utf-8"
- Another `<meta>` tag with the attribute "name" and the value "viewport," as well as a "content" attribute with a value of "width=device-width" (This will be used for responsive web design later.)

This is the expected input and output (output should be blank):

```
index.html •
<!DOCTYPE html>
<html>
  <head>
    <title>MyFirstPage - Home</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width">
  </head>
  <body></body>
</html>
```

5. Next, add the following semantic tags to the body section, making sure to properly nest the main and aside elements within the section element:

- a. <header>
- b. <nav>
- c. <section>
  - i. <main>
  - ii. <aside>
- d. <footer>

This is the expected input:

```
index.html •
<!DOCTYPE html>
<html>
  <head>
    <title>MyFirstPage - Home</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width">
  </head>
  <body>
    <header></header>
    <nav></nav>
    <section>
      <main></main>
      <aside></aside>
    </section>
    <footer></footer>
  </body>
</html>
```

6. Place a copy of the name of each semantic section within the semantic tags like so: <header>Header</header>.

This is the expected input:

```
index.html X P
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <title>MyFirstPage - Home</title>
6   <meta charset="utf-8">
7   <meta name="viewport" content="width=device-width">
8 </head>
9
10 <body>
11   <header>Header</header>
12   <nav>Nav</nav>
13   <section>Section
14     <main>Main</main>
15     <aside>Aside</aside>
16   </section>
17   <footer>Footer</footer>
18 </body>
19
20 </html>
21
```

7. Since semantic tags do not have any default visual impact, the only thing you should see in the rendered output is the names of the semantic sections stacked on top of each other. Don't worry that the "aside" element is not positioned to the left or right of the "main" element; we will tackle this in the next challenge using CSS.

This is the expected input with the output:

```
index.html X P [Icons] stackblitzstartersxq37qe-bxuz--8080-...
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <title>MyFirstPage - Home</title>
6   <meta charset="utf-8">
7   <meta name="viewport" content="width=device-width">
8 </head>
9
10 <body>
11   <header>Header</header>
12   <nav>Nav</nav>
13   <section>Section
14     <main>Main</main>
15     <aside>Aside</aside>
16   </section>
17   <footer>Footer</footer>
18 </body>
19
20 </html>
21
```

Header  
Nav  
Section  
Main  
Aside  
Footer

8. If you haven't already done so, now is a good time to save your project.

## REFLECT

This core structure of a webpage will be the same core structure that you will use throughout your career as a web developer. The only things that you might modify are the semantic elements in the body as you may need to add or remove sections per the client's needs.

## DID YOU KNOW

The source code for public websites is available for users to examine. When you visit a website, such as [www.bettycrocker.com](http://www.bettycrocker.com), and right-click anywhere on the page (other than on a picture), you can select “View Source Code” or “Inspect” to view the source code for the website. “View Source Code” shows you the original HTML document as it exists on the server and opens it in a new tab. Notice that the very first line in the source code is the `<!DOCTYPE>` tag.


The “Inspect” tool also shows you the interactive and live source code for the webpage. Keep in mind that this code is live, and the HTML DOM has likely been modified by the browser, JavaScript, or plugins. To see the original code as it exists on the server, use the “View Source Code” option.

A final note about the Inspect option: You can edit the live HTML DOM using this tool, and you can see the results of the change appear in the browser. This is only temporary since reloading the page will overwrite your changes with the original webpage.

Take a few minutes to explore. What tags do you recognize?

## 2. Common Tags

In addition to the tags covered previously, we will discuss some of the common tags used in the body section of a webpage.

Name	Tag	Description
Paragraph	<code>&lt;p&gt;</code>	The <b>paragraph tag</b> is used to surround blocks of text that should be formatted as a paragraph. The main effect of using the paragraph tag is that it ensures a paragraph is preceded and followed by a blank line. Additionally, as with any tag, a paragraph could be given an id or class attribute in order to stylize the paragraph of text.
Heading	<code>&lt;h1&gt;</code> <code>&lt;h2&gt;</code> <code>&lt;h3&gt;</code> <code>&lt;h4&gt;</code> <code>&lt;h5&gt;</code> <code>&lt;h6&gt;</code>	The <b>heading tag</b> is used to create textual headings to help organize content. Heading levels range from <code>&lt;h1&gt;</code> to <code>&lt;h6&gt;</code> , with <code>&lt;h1&gt;</code> being the largest heading. All webpages require one and only one <code>&lt;h1&gt;</code> heading. Headings should also be used sequentially (you should not skip heading levels). Thus, after the <code>&lt;h1&gt;</code> heading, the next heading would be <code>&lt;h2&gt;</code> , the next could be another <code>&lt;h2&gt;</code> or an <code>&lt;h3&gt;</code> , and the next one after that can be an <code>&lt;h2&gt;</code> , <code>&lt;h3&gt;</code> , or <code>&lt;h4&gt;</code> . Think of a document outline structure.
Division	<code>&lt;div&gt;</code>	The <b>division tag</b> is used to create a division or section of content anywhere on a webpage. Essentially, a <code>&lt;div&gt;</code> element creates a block of space related to the division element and its contents. Dives do not contain any visual elements by default, other than its dimensions (height and width). However, <code>&lt;div&gt;</code> elements are often used with an id or class in order to provide CSS formatting and layout properties. Most commonly, <code>&lt;div&gt;</code> elements are used in conjunction with Flexbox or the Bootstrap framework to provide layout and responsiveness mechanisms.
Table	<code>&lt;table&gt;</code>	<div>The <b>table tag</b>, which will be discussed more later in this tutorial, is used to define a table of data and to hold tabular data. Table elements contain additional tags to define the rows of the table and the number of cells within each row. As an example, tables are used to display hours of operation, restaurant menus, pricing scales, etc.</div> <div> <b>KEY CONCEPT</b></div> <div>It may be tempting to use tables in place of semantic tags to provide a layout structure for your entire page. However, this is bad practice as it has a significant negative impact on your</div>

		site's SEO score and does not lend itself to responsive web design.
Unordered List	<ul>	The <b>unordered list tag</b> provides a bulleted list of items, words, statements, etc. The <ul> element is the container of the <li> list item elements. The bullets have a few variations, such as a solid dot (default), an empty circle, an empty disk or ellipse, and nothing. You can even use a custom image for a bullet. Unordered lists are used when the list of items is not sequential or progressive. Unordered lists, in conjunction with CSS, are also commonly used as the base for building navigation menus.
Ordered List	<ol>	<p>The <b>ordered list tag</b> works the same as an unordered list; the difference is that the items in an ordered list are sequential and have a specific order, much like the instructions for baking a pie. Each item is numbered, instead of bullets, and the format of the numbers can be modified to be a wide range of sequential numbers from different languages.</p> <p>While there are 28 different ordered list types, the common types used include decimal (default), lower Roman numerals, upper Roman numerals, lower Alpha character, and upper Alpha character. The common ordered list types can be easily accessed using an HTML attribute "type" and then providing one of the following values respectively: "1," "i," "I," "a," or "A".</p>
List Item	<li>	The <b>list item tag</b> is used to create each item in an ordered or unordered list. The <li> tags are nested within the <ol> or <ul> elements, and each item in the list is a complete <li></li> element. List items only have one HTML attribute, "value," and are only used within ordered lists to indicate the first item's starting value. For example, if you wanted a second list on a page to pick up counting where the first list left off, you would give the first list item in the second list the value attribute and set it to the desired starting value.
Horizontal Rule	<hr>	The <b>horizontal rule</b> is a self-closing tag that creates a visual line horizontally across the page. This line appears anywhere you place the <hr> tag. If you wanted to visually separate the page's subheading and the main paragraphs of text with a horizontal rule, you would close the subtitle's <p> element, add the <hr> tag, and then begin the next <p> element.
Line Break	 	<p>The <b>line break tag</b> or break tag is a self-closing tag used to create a line break in the page content. Adding a break in the middle of a paragraph element will split the paragraph in two.</p> <p>🔗 EXAMPLE</p> <p>&lt;p&gt; This sentence with a &lt;br&gt; line break in the middle of it. &lt;/p&gt;</p> <p>This will be displayed as follows:</p> <p>This sentence with a line break in the middle of it.</p>
Span	<span>	The <b>span tag</b> is an inline element that can be used to surround a section of text or other content in order to give it an id or class value. Spans do not change the layout or any other visual aspects of the contained content unless specified by the stylesheet.



We have briefly covered most of the basic HTML tags commonly used in web design. For a complete list of HTML tags, visit [www.w3schools.com/tags](https://www.w3schools.com/tags).



View the following video for more on creating your first HTML document. You can follow along by returning to your StackBlitz account and creating a new project using the HTML, CSS, and JavaScript template.



## TERMS TO KNOW

### Paragraph Tag <p>

An HTML tag used to define a paragraph of text.

### Heading Tag <h#>

An HTML tag used to create textual headings to help organize content.

### Division Tag <div>

An HTML tag used to create a division or block of content.

### Table Tag <table>

An HTML structure that is designed to hold and organize tabular data.

### Unordered List Tag <ul>

An HTML tag used to create bulleted, nonsequential lists of items.

### Ordered List Tag <ol>

An HTML tag used to create sequenced numbered lists of items.

### List Item Tag <li>

An HTML tag for creating each item in either an ordered or unordered list.

### Horizontal Rule <hr>

A self-closing HTML tag that creates a visual line horizontally across the page.

### Line Break Tag <br>

An HTML tag used to create a non-paragraph-breaking line break.

### Span Tag <span>

An HTML tag used to surround inline content to apply styles to a section of content.



## SUMMARY

In this lesson, you learned about the **core code structure** of an HTML webpage and how it should be organized. You also learned what each part of the core structure does, including the doctype statement, the <html> element, and its <head> and <body> sections. Then, you learned about the **common tags** used in developing the content of a webpage. With time, practice, and research, you will become familiar with these common tags' various uses and abilities, as you will be using them throughout most of the webpages that you will develop.

Source: This Tutorial has been adapted from "The Missing Link: An Introduction to Web Development and Programming " by Michael Mendez. Access for free at <https://open.umn.edu/opentextbooks/textbooks/the-missing-link-an-introduction-to-web-development-and-programming>. License: **Creative Commons attribution: CC BY-NC-SA**

**Body Tag <body>**

An HTML tag used to define the body of content of a digital document, which is used to create all of the visible content that will be displayed on the page.

**Character Set**

A system of numbers used to encode text-based characters for storage and retrieval of text data within a computer file. Also called *character encoding*.

**DOCTYPE Declaration <!DOCTYPE>**

A special tag used to indicate the type and format of the digital document.

**Division Tag <div>**

An HTML tag used to create a division or block of content.

**Head Tag <head>**

An HTML tag used to define the head portion of a webpage, which is used to configure the page and link to resources in external files.

**Heading Tag <h#>**

An HTML tag used to create textual headings to help organize content.

**Horizontal Rule <hr>**

A self-closing HTML tag that creates a visual line horizontally across the page.

**JavaScript Functions**

A named set of instructions that can accept data as well as return an output value, written in JavaScript, and can be called as many times as possible, whenever needed.

**Line Break Tag <br>**

An HTML tag used to create a non-paragraph-breaking line break.

**Link Tag <link>**

An HTML tag used to link to either an external stylesheet or an icon image file for the webpage.

**List Item Tag <li>**

An HTML tag for creating each item in either an ordered or unordered list.

**Ordered List Tag <ol>**

An HTML tag used to create sequenced numbered lists of items.

**Paragraph Tag <p>**

An HTML tag used to define a paragraph of text.

**Script Tag <script>**

An HTML tag used to embed JavaScript code directly within an HTML document.

**Span Tag <span>**

An HTML tag used to surround inline content to apply styles to a section of content.

**Style Tag <style>**

An HTML tag used to embed a section of CSS style rules within the <head> section of an webpage that only affects the page in which it is contained.



**Table Tag <table>**

An HTML structure that is designed to hold and organize tabular data.

**Title Tag <title>**

An HTML tag that goes in the <head> section of a webpage that sets the title value of the web browser tab.

**Unordered List Tag <ul>**

An HTML tag used to create bulleted, nonsequential lists of items.