PER SCHOLAS

Fundamentals of HTML

Learning Objectives

This lesson provides an overview of the basic and fundamental topics of the HTML and CSS.
By the end of this lesson, learners will be able to:

- Create an HTML page.
- Identify important elements in HTML structure.
- Describe the parent-to-child relationship between elements.
- Create elements and assign them attributes.
- Inspect elements within a webpage using the browser inspector.
- Use the Emmet Toolkit to quickly create HTML elements.

PER SCHOLAS

Table of Contents

PER SCHOLAS

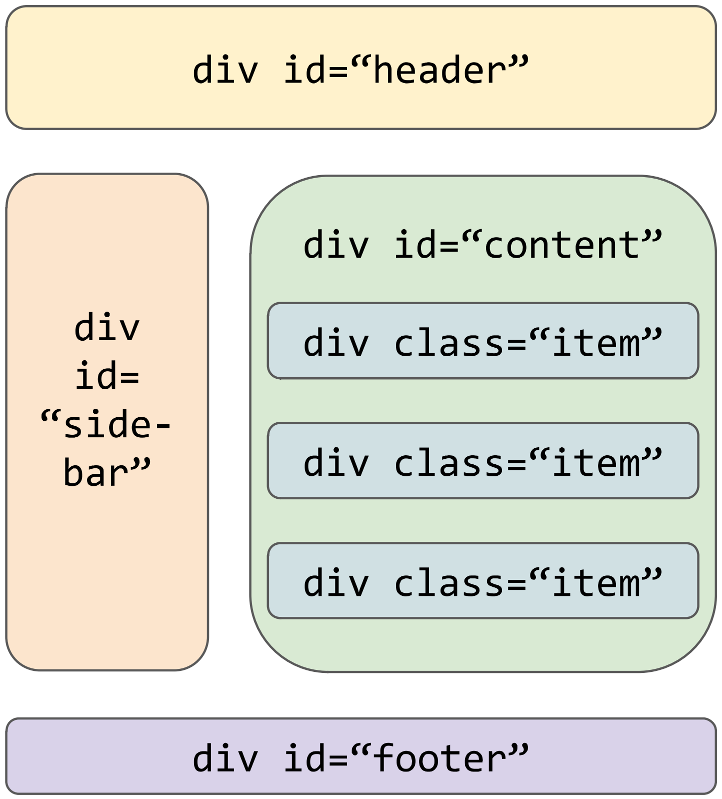
Section 1
Basic HTML

PER SCHOLAS

HyperText Markup Language

- **HTML (Hypertext Markup Language)** is a markup language used to design web pages.
- HTML allows the creation of web content, and tells the browser how to display the content.
- HTML structures paragraphs, sections, and links by using elements (the building blocks of a web page).
- The two most commonly used extensions of HTML documents are **.html** and **.htm**.
- **HTML5** is the latest version of the HTML specification.
- **UTF-8** is the default character encoding for HTML5.

Example: Basic HTML Page Layout



Source: Per Scholas

PER SCHOLAS

HTML Elements Definition

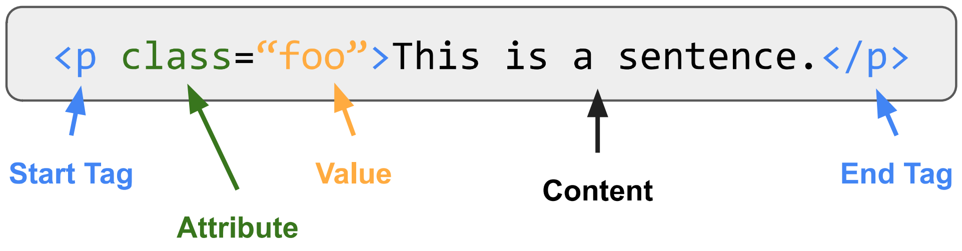
An HTML element helps the browser determine what default structure and style content the screen should display.

Elements and tags are not the same. Tags begin or end an element in source code, whereas elements are part of the DOM, the document model for displaying the page in the browser.

Examples of HTML tags include:

- `<html></html>`
- `<head></head>`
- `<body></body>`
- `<div></div>`
- `<p></p>`
- `<img></img>`

HTML Layout Breakdown



```
<p class="foo">This is a sentence.</p>
```

An HTML element consists of an opened and closed HTML tag, while some tags are self-closed:

- Opening tag: <nav>
- Closing tag: </nav>
- Self-closed tag: <img />

[Click here for reference documentation on HTML elements.]

PER SCHOLAS

HTML Base Template Code

`<!DOCTYPE>`

When performing HTML validation testing on a web page, the `<!DOCTYPE>` element tells the HTML validator which version of HTML standards the web page coding is to comply with.

When validating the web page, the HTML validator checks the coding against the applicable standard, and then reports which portions of the coding do not pass HTML validation.

`<html>`

The `<html>` element tells the browser that this is an HTML document, and it represents the root of an HTML document. The `<html>` tag is the container for all other HTML elements (except for the `<!DOCTYPE>` tag).

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Title</title>
    </head>
    <body>

    </body>
</html>
```

PER SCHOLAS

HTML Base Template Code (continued)

`<head>`

The `<head>` element is a container for metadata (data about data), and is placed between the `<html>` tag and the `<body>` tag.

- ➢ Metadata is not displayed.
- ➢ HTML metadata is data about the HTML document.
- ➢ Metadata typically defines the document title, character set, styles, links, scripts, and other meta information.

`<title>`

The `<title>` element sets a title to the browser page.

`<body>`

The `<body>` element is where all visible content being displayed in a web page is located within the HTML document.

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Title</title>
    </head>
    <body>

    </body>
</html>
```

PER SCHOLAS

Parent-to-Child Relationship

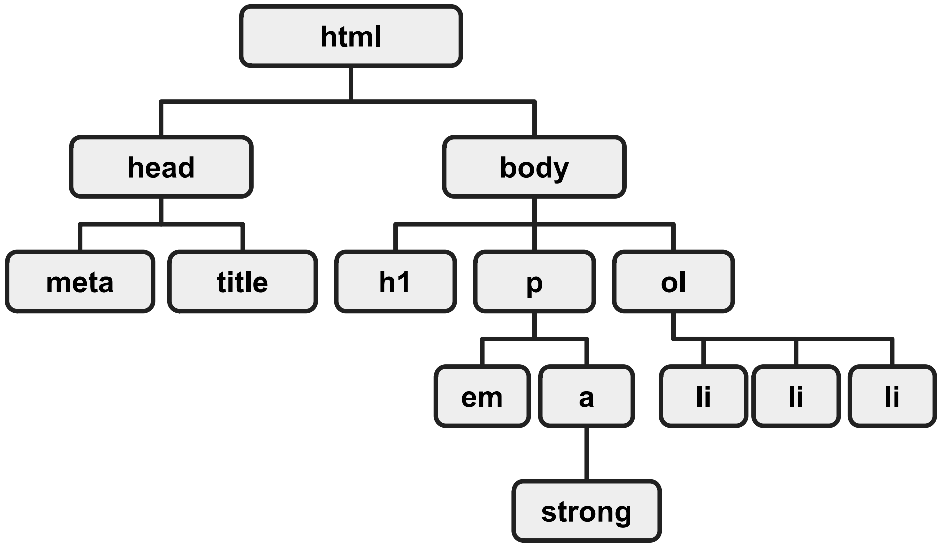
The document tree becomes a family tree, which refers to the relationship between elements.

- An element that is directly contained by another element is said to be the child of that element.
- The Inspector makes it very easy to see which tags are parent to other tags.
- Understanding the parent-to-child relationship is important because:
  - It tells you how the HTML is structured.
  - It plays a big role when you are ready to apply style to your content.

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>This Title</title>
    </head>
    <body>
        <h1>Main Header</h1>
        <p>
            Lorem ipsum <em>dolor</em> sit amet,
            <a href="#">
                Consectetur <strong>adipisicing</strong> elit
            </a>
            , sed do eiusmod
            tempor incididunt ut labore
        </p>
        <ol>
            <li>Point One</li>
            <li>Point One</li>
            <li>Point One</li>
        </ol>
    </body>
</html>
```

PER SCHOLAS

Example: Parent-to-Child Relationship



```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>This Title</title>
    </head>
    <body>
        <h1>Main Header</h1>
        <p>
            Lorem ipsum <em>dolor</em> sit amet,
            <a href="#">
                Consectetur <strong>adipisicing</strong> elit
            </a>
            , sed do eiusmod
            tempor incididunt ut labore
        </p>
        <ol>
            <li>Point One</li>
            <li>Point One</li>
            <li>Point One</li>
        </ol>
    </body>
</html>
```

PER SCHOLAS

HTML Word-Wrap and Whitespace

Most of us are accustomed to typing text in a word processor, and letting the program determine where the line breaks belong. This is known as *word-wrap*. The only time that we are required to hit the enter key is when we want to start a new paragraph.

Browsers, however, will use word-wrap to display text, even if the enter key is pressed.

- ➢ Browsers will treat a new line character, a tab character, and multiple spaces as a single space. If there are five spaces at the start of a line, they will be compressed into one space.
- ➢ In order to insert a new line, tab, or multiple space in an HTML page, markup must be used.
- ➢ If it is not plain text, it must be placed in markup.

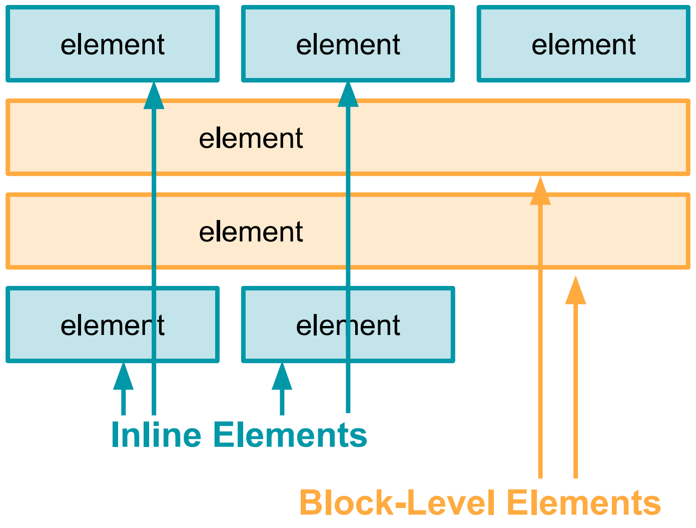
PER SCHOLAS

Block-Level Elements in HTML

Block-level elements form a visible block on a page — they will appear on a new line from whatever content went before them, and any content that goes after it will also appear on a new line. Block-level elements tend to be structural elements on the page that represent, for example, paragraphs, lists, navigation menus, footers, etc. A block-level element would not be nested inside an inline element, but it might be nested inside another block-level element.

Examples of Block-Level Elements in HTML

`<address>`	`<dd>`	`<figcaption>`	`<header>`	`<noscript>`	`<table>`
`<article>`	`<div>`	`<figure>`	`<hr>`	`<ol>`	`<tfoot>`
`<aside>`	`<dl>`	`<footer>`	`<li>`	`<p>`	`<ul>`
`<blockquote>`	`<dt>`	`<form>`	`<main>`	`<pre>`	`<video>`
`<canvas>`	`<fieldset>`	`<h1> - <h6>`	`<nav>`	`<section>`	



PER SCHOLAS

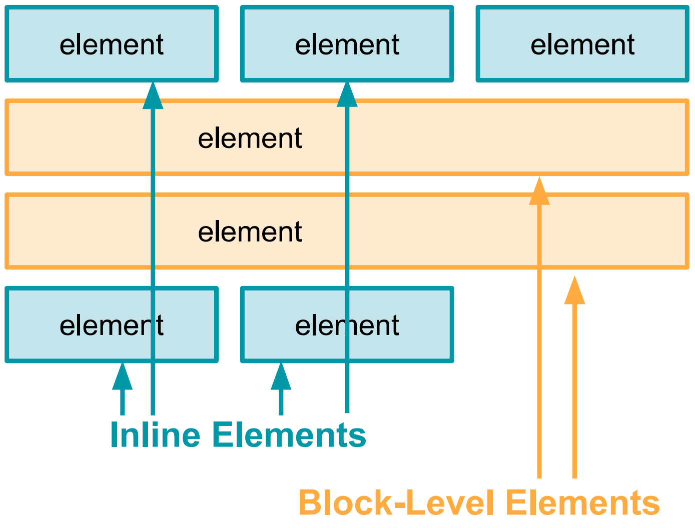
Inline Elements in HTML

Inline elements are those that are contained **within block-level elements** and surround only small parts of the document's content; not entire paragraphs or groupings of content.

An inline element will **not** cause a new line to appear in the document; they would normally appear inside a paragraph of text (e.g., a hyperlink element **`<a>`**, or emphasis elements such as **`<em>`** or **`<strong>`**).

Examples of Inline Elements in HTML

`<a>`	`<big>`	`<dfn>`	`<kbd>`	`<q>`	`<span>`
`<abbr>`	`<br>`	`<em>`	`<label>`	`<samp>`	`<strong>`
`<acronym>`	`<button>`	`<i>`	`<map>`	`<script>`	`<sub>`
`<b>`	`<cite>`	`<img>`	`<object>`	`<select>`	`<sup>`
`<bdo>`	`<code>`	`<input>`	`<output>`	`<small>`	`<textarea>`



PER SCHOLAS

Void Elements and Self-Closing Tags

Void elements are elements that do not have or require a closing tag because they cannot have any child nodes.

It is important to note that self-closing tags do not actually exist in HTML. If a trailing slash character is present in the start tag of an HTML element, HTML parsers ignore that slash character.

However, self-closing tags are required in void elements in XML, XHTML, and SVG, making it good practice to self-close tags for readability and compatibility in HTML code.

There are very few void elements in HTML:

`<area>`	`<embed>`	`<link>`	`<wbr>`
`<base>`	`<hr>`	`<meta>`	
`<br>`	`<img>`	`<source>`	
`<col>`	`<input>`	`<track>`	

PER SCHOLAS

HTML Attributes

Some tags require an attribute to function properly. For example, an `<img>` tag requires a `src` (source) attribute and an `alt` (alternate description) attribute for it to validate.

```
<img source="./some/source/location" alt="An image description." />
```

Attributes are extra information you can give to an element. In the case of the `<img>` tag, the `<src>` attribute provides a way for the `<img>` to source an image from your local computer or from the internet.

An attribute should have:

- ➢ A space between the element name and the attribute name.
- ➢ The attribute name, followed by an equals (=) sign.
- ➢ An attribute value, with opening and closing quotation marks wrapped around it.
- ➢ **In HTML, attributes usually come in name/value pairs, such as `name="value"`.**

Click here for a full reference on HTML attributes.

PER SCHOLAS

Knowledge Check

- ➢ What is HTML, and what is it used for?
- ➢ What is the default character set used for HTML5?
- ➢ What is the difference between an element and a tag in HTML?
- ➢ What base template elements should web pages typically contain?
- ➢ Which tag contains all of the visible content of an HTML document?
- ➢ What is the parent-to-child relationship between elements and how is it represented in code?
- ➢ How do browsers handle whitespace in HTML documents?
- ➢ What is the difference between block-level elements and inline elements?
- ➢ What are void elements (also referred to as self-closing tags)?
- ➢ What are element attributes?

PER SCHOLAS

Section 2
HTML Elements

PER SCHOLAS

HTML Heading Element

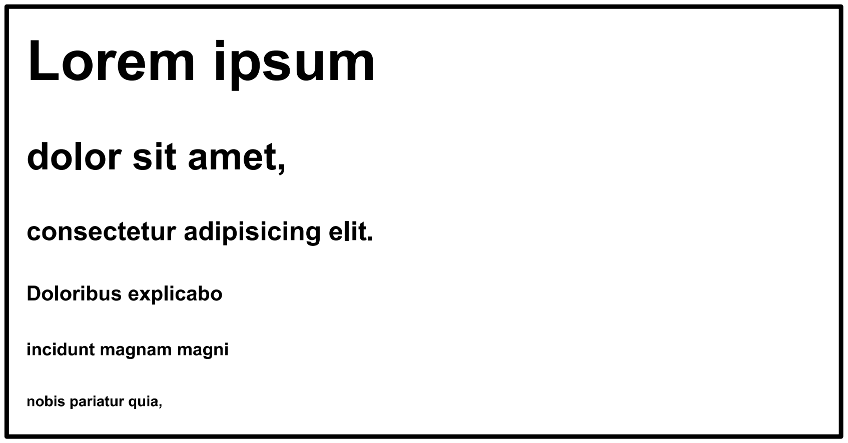
`<h1>, <h2>, <h3>, <h4>, <h5>, <h6>`

The `<h1>-<h6> HTML elements` specify headings with relative levels of importance, which is reflected in the formatting of the text when the HTML page is rendered. **`<h1>`** elements have the highest importance, and **`<h6>`** elements have the lowest importance.

You can see how these tags affect their associated text by visiting this reference page on HTML headings.

```
<!DOCTYPE html>
<html>
    <body>
        <h1>Lorem ipsum</h1>
        <h2>dolor sit amet,<h2>
        <h3>consectetur adipisicing elit.<h3>
        <h4>Doloribus explicabo<h4>
        <h5>incidunt magnam magni<h5>
        <h6>nobis pariatur quia,<h6>
    </body>
</html>
```

Result



Lorem ipsum

dolor sit amet,

consectetur adipisicing elit.

Doloribus explicabo

incidunt magnam magni

nobis pariatur quia,

PER SCHOLAS

HTML Paragraph Element

`<p>`

The <p> HTML element defines a paragraph. A paragraph always starts on a new line, and browsers automatically add some white space (a margin) before and after a paragraph.

```
<!DOCTYPE html>
<html>
    <body>
        <p>Lorem ipsum dolor sit amet, consectetur adipisicing
elit.</p>
        <p>Doloribus explicabo incidunt magnam magni nobis
pariatur quia, rerum sint tempora vero.</p>
    </body>
</html>
```

```
Lorem ipsum dolor sit amet, consectetur adipisicing elit.

Doloribus explicabo incidunt magnam magni nobis pariatur quia,
rerum sint tempora vero.
```

Result

PER SCHOLAS

HTML Line Break and Thematic Break Elements

`<br>`

The `<br>` HTML element produces a line break in text (also known as a carriage-return). It is useful for writing where the division of lines is significant (e.g., a poem or address).

`<hr>`

The `<hr>` HTML element produces a thematic break in text (also known as a horizontal rule).

```html
<!DOCTYPE html>
<html>
    <body>
        <p>Lorem ipsum dolor <br>
            sit amet, consectetur adipisicing elit. <br>
            <hr>
            Doloribus explicabo incidunt magnam magni nobis <br>
            pariatur quia, rerum sint tempora vero. <br>
      </p>
    </body>
</html>
```

Lorem ipsum dolor
sit amet, consectetur adipisicing elit.

Doloribus explicabo incidunt magnam magni nobis
pariatur quia, rerum sint tempora vero.

Result

PER SCHOLAS

HTML Preformatted Text Element

`<pre>`

The `<pre>` HTML element represents preformatted text, which is to be presented exactly as written in the HTML file. The text is typically rendered using a non-proportional, or monospaced font. Whitespace inside this element is displayed as written.

```
<!DOCTYPE html>
<html>
    <body>
        <pre>Lorem ipsum dolor
sit amet, consectetur adipisicing elit.
Doloribus explicabo incidunt magnam magni nobis
pariatur quia, rerum sint tempora vero.
       </pre>
    </body>
</html>
```

```
Lorem ipsum dolor
sit amet, consectetur adipisicing elit.
Doloribus explicabo incidunt magnam magni nobis
pariatur quia, rerum sint tempora vero.
```

Result

HTML Text Formatting Elements

HTML contains several elements for formatting text in specific ways:

- `<b>` - Bold
- `<i>` - Italic
- `<strong>` - Important
- `<em>` - Emphasized
- `<mark>` - Highlighted
- `<small>` - Small
- `<del>` - Deleted
- `<ins>` - Inserted
- `<sup>` - Superscript
- `<sub>` - Subscript

```html
<!DOCTYPE html>
<html>
    <body>
        <p>
            <b>This text is bold.</b><br>
            <i>This text is italic.</i><br>
            <strong>This text is important!</strong><br>
            <em>This text is emphasized.</em><br>
            This text is <mark>highlighted</mark>.<br>
            <small>This text is small.</small><br>
            <del>This text is deleted.</del><br>
            <ins>This text is inserted.</ins><br>
            This text is <sup>superscripted</sup>.<br>
            This text is <sub>subscripted</sub>.<br>
        </p>
    </body>
</html>
```

Result

This text is bold.
This text is italic.
This text is important!
This text is emphasized.
This text is marked.
This text is small.
~~This text is deleted.~~
This text is inserted.
This text is $^{\text{superscripted}}$.
This text is $_{\text{subscripted}}$.

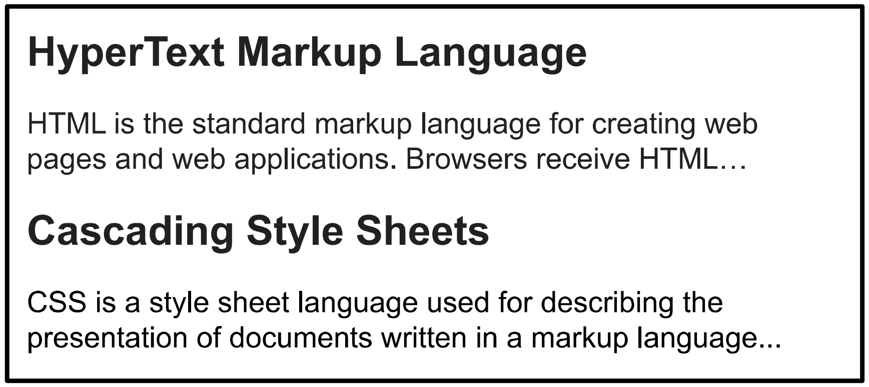
PER SCHOLAS

HTML Section Element

`<section>`

The `<section>` HTML element represents a standalone section, which does not have a more specific semantic element to represent it. The **`<section>`** element is most frequently used to group together related elements. Each **`<section>`** typically includes one or more heading elements and additional elements presenting related content.

```
<!DOCTYPE html>
<html>
    <body>
        <section>
            <h2>HyperText Markup Language</h2>
            <p>HTML is the standard markup language for creating
web pages and web applications. Browsers receive HTML...</p>
        </section>
        <section>
            <h2>Cascading Style Sheets</h2>
            <p>CSS is a style sheet language used for describing
the presentation of documents written in a markup language...</p>
    </section>
    </body>
</html>
```



HyperText Markup Language

HTML is the standard markup language for creating web pages and web applications. Browsers receive HTML…

Cascading Style Sheets

CSS is a style sheet language used for describing the presentation of documents written in a markup language...

Result

PER SCHOLAS

HTML List Elements

Lists in HTML come in two varieties: ordered lists and unordered lists. Ordered lists are typically rendered with numbers next to the list elements, whereas unordered lists are typically rendered with bullet points. Lists can be nested within one another, and can mix between ordered and unordered lists.

`<ol>`

The `<ol>` HTML element represents ordered lists, in which the order of the list elements is meaningful. The `<ol>` element takes a `type` attribute that can be used to change the characters used to order the list.

`<ul>`

The `<ul>` HTML element represents unordered lists, in which the order of the list elements is not meaningful.

`<li>`

The `<li>` HTML element represents list items, and is used to define each child element within an ordered or unordered list.

```
<!DOCTYPE html>
<html>
    <body>
        <h1>List Example</h1>
        <ol>
            <li>Item One</li>
            <li>Item Two</li>
            <ul>
                <li>Item Two-One</li>
                <li>Item Two-Two</li>
            </ul>
            <li>Item Three</li>
        </ol>
    </body>
</html>
```

List Example

1. Item One
2. Item Two
   - Item Two-One
   - Item Two-Two
3. Item Three

Result

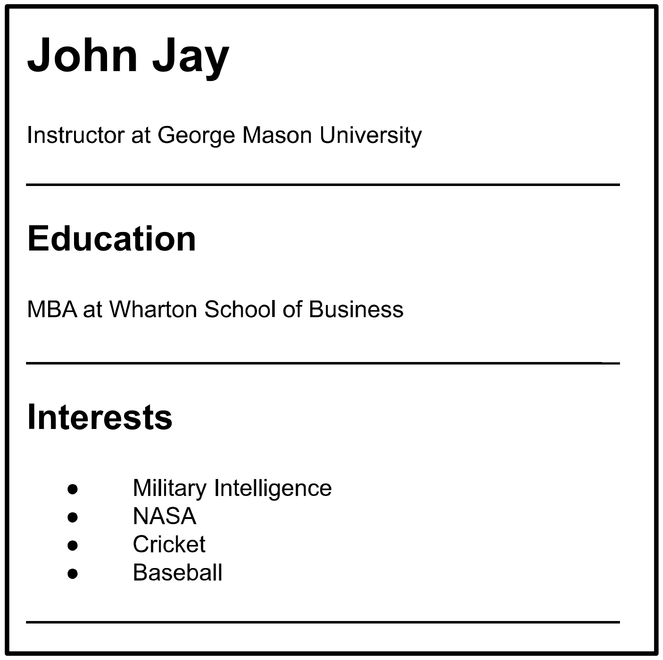
Practice Assignment: Basic HTML Layout

Write an HTML document showing your resume using **`<br>`** tags, **`<hr>`** tags, and other HTML Elements.

Sample Code

```html
<!DOCTYPE html>
<html>
    <body>
        <h1>John Jay</h1>
        Instructor at George Mason University
        <hr>
        <h2>Education</h2>
        MBA at Wharton School of Business
        <hr>
        <h2>Interests</h2>
        <ul>
            <li>Military Intelligence</li>
            <li>NASA</li>
            <li>Cricket</li>
            <li>Baseball</li>
        </ul>
        <hr>
    </body>
</html>
```

Sample Result



John Jay

Instructor at George Mason University

Education

MBA at Wharton School of Business

Interests

- Military Intelligence
- NASA
- Cricket
- Baseball

PER SCHOLAS

HTML Image Element

`<img>`

The [`<img>` HTML element] is used to insert an image into an HTML document. The image itself is not inserted directly into the document; instead, the browser inserts an HTML image from the source specified in the `<img>` element.

There are two required attributes for an `<img>` element:

- ➢ **`src`** - used to show the image source.
- ➢ **`alt`** - defines an alternate text description for the image.

Optional attributes include **`height`**, **`width`**, and **`border`**.

Example Image Element

```
<img source="./some/source/location” alt=“An image description.” height=“500px” width=“500px” />
```

PER SCHOLAS

HTML Anchor Element

`<a>`

The <a> HTML element is used to reference or link to other resources, such as HTML documents and images. Web browsers typically underline text hyperlinks and color them blue by default.

There is one required attribute for an `<a>` element:

- ➢ **`href`** - used to specify the URL that the anchor element points to.

Another important attribute is **`target`**. The **`target`** attribute specifies where to open the linked document, and has a default value of **`_self`**, which opens the document in the same browser tab/window as the link is located.

Other values for the **`target`** attribute are:

- ➢ **`_blank`** - opens the document in a new window or tab.
- ➢ **`_parent`** - opens the document in the parent frame.
- ➢ **`_top`** - opens the document in the full body of the window.

Example Anchor Element

```
<a href="https://perscholas.org/" target="_blank">Head to the Per Scholas Website!</a>
```

PER SCHOLAS

HTML Anchor Element: Email and Phone Linking

You may need to open the user’s default email client with an email address when users click on an anchor link. You can do this by using the **`mailto`** protocol as part of the **`href`** attribute's value.

The syntax of the value should be in the form of **`mailto:email@address.com`**.

```
<a href=“mailto:me@example.com”>Send me an email!</a>
```

Similarly, you can include a phone number, which will be opened in the user’s default phone app when the link is clicked. You can do this by using the **`tel`** protocol as part of the **`href`** attribute value.

The syntax of the value should be in the form of **`tel:+##########`**.

```
<a href=“tel:+15555551234”>Give me a call!</a>
```

Activity: Images and Hyperlinks

Write an HTML program with images and hyperlinks.

- ➢ Download the logos of a few large companies, like Google, Facebook, and Apple.
- ➢ Write an HTML file that includes those logos in **`<img>`** tags.
  - ○ Use **`height="200"`** and **`width="300"`** as initial values for the images.
  - ○ Change the height and width so that images have a good look and feel.
- ➢ Provide the names of the logos in text under each image, and within the **`<img>`** tag's **`alt`** attribute.
- ➢ Create an image hyperlink by nesting the **`<img>`** elements in anchor elements for all of the logos, such that if the user clicks on the Google logo it will take them to www.google.com.
- ➢ Experiment with adding additional content, and refining the layout.

PER SCHOLAS

Section 3
Organization

PER SCHOLAS

HTML Comments

In HTML, a comment is a section of text that is not processed by the web browser. Comments are enclosed in `<!-- -->` tags. These tags tell the browser that the text inside them is a comment and should not be rendered on the front end.

Comments are important to keep your code organized and readable.

Example HTML Comments

```
<!DOCTYPE html>
<html>
    <body>
        <h1>Lorem ipsum</h1>

        <!-- Sample HTML Comment -->
        <!-- These two lines will not be displayed. -->

        <p>
            Lorem ipsum <em>dolor</em> sit amet,
        </p>
    </body>
</html>
```

PER SCHOLAS

Folder Structure

The folder structure of an application is very important for organization and functionality, and should be well-defined.

When working on a website locally, keep all related files in a single folder on the server that mirrors the published website's file structure. This folder can live anywhere, but should be located somewhere that it can be easily found (e.g., desktop, a project folder, or the root of the hard drive).

On the following slide, we will create a template folder structure for front-end work.

PER SCHOLAS

Activity: Folder Structure

In your Documents, create a folder named **Front-End-Work.**
Within that folder, create the following additional folders:

- **pages**: This folder will contain all HTML files.
    - Using Visual Studio Code or your IDE of choice, create a new file called: **index.html**, and save it inside your **pages** folder. This file will generally contain homepage content; that is, the text and images that people see when they first go to the website.
- **images**: This folder will contain all of the images that you use on your site.
- **styles**: This folder will contain the CSS code used to style your content (e.g., setting text and background colors).

PER SCHOLAS

Absolute and Relative Paths

The file path indicates the location of a file.

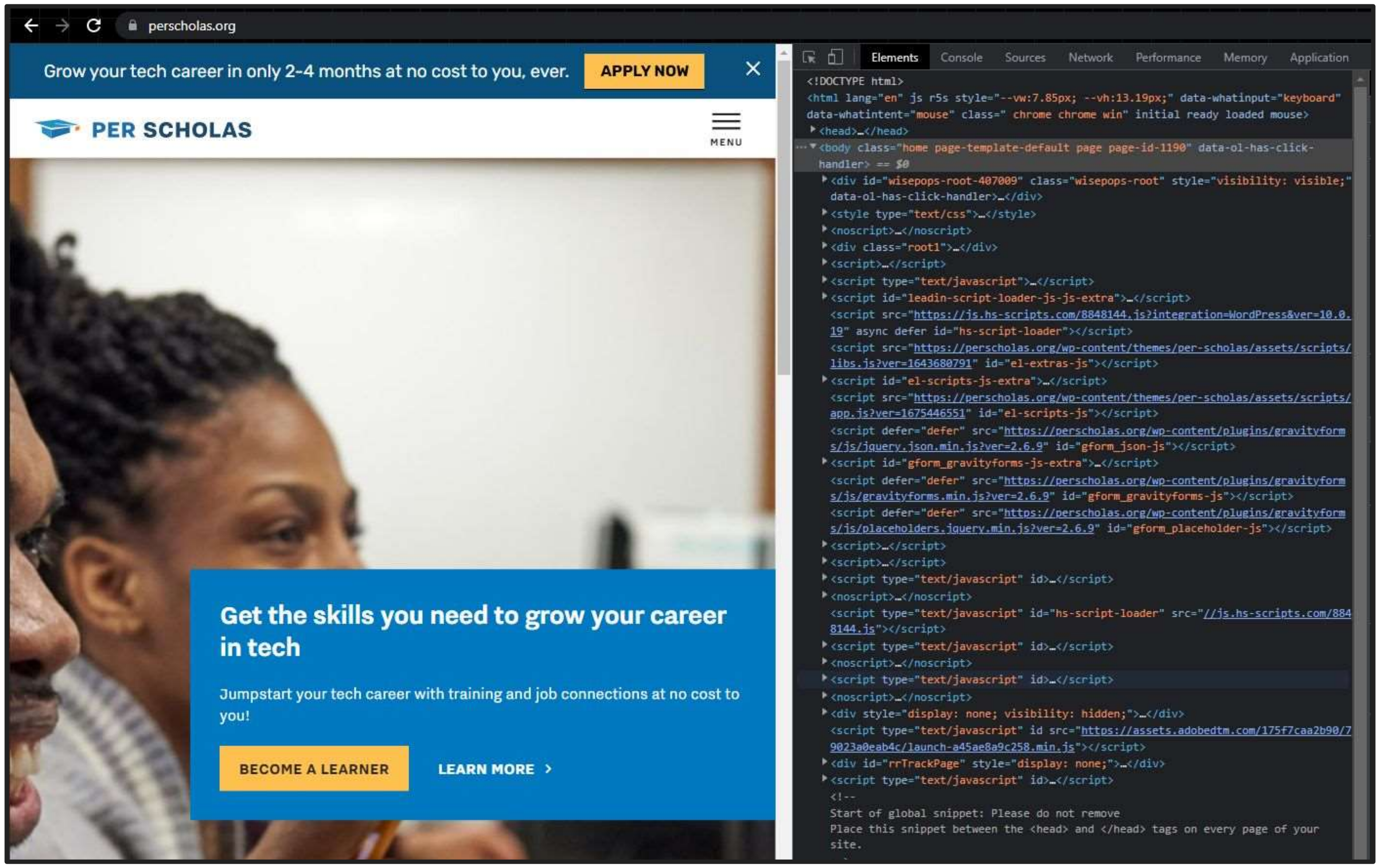
- **Absolute Path:** An absolute path shows the complete path to a file starting from the web root. A file in your computer would have an absolute path starting from the C: drive, such as "`C:\Users\Learner\Desktop`."
- **Relative Path:** Relative paths can only be used to link to other files under the same web root. A relative path indicates where a file is located relative to the file that contains the link.
- The properties of a path include:
  - **"`.`" (period) -** indicates the current working directory.
  - **"`..`" (two periods) -** indicates to go up one directory from the current directory.
  - **"/" (forward slash) -** separates directories. For instance, imagine three directories, dir0, dir1, and dir2, where dir0 contains dir1 and dir1 contains dir2. The path from dir0 to dir2 (assuming that the current working directory is dir0) is "**`./dir1/dir2`**."
    - Note that Windows uses the **backslash "\"** to indicate absolute paths within directories, as shown in the absolute path example above, but the vast majority of programming languages use forward slashes in all paths.

Browser Inspector

- On Google Chrome, navigate to: https://perscholas.org/
- There are a few options to open the **Inspector**:
  - Right-click on the page and select “Inspect” or “Inspect Element.”
  - Press “Ctrl + Shift + C” on Windows or “Command + Option + C” on Mac.
  - In Chrome, press F12.
- In the inspector, you are able to see the entire structure of a single HTML page, including scripts, style sheets, or other content.
- In the **Console** tab, you are able to run JavaScript, which talks only to the browser. This will be discussed in more detail in the JavaScript lessons.
- In the **Style** tab, you are able to see the style applied to a particular element. This will be discussed in more detail in the Cascading Style Sheets (CSS) lessons.
- In the **Elements** tab, you are able to see every HTML element that is being used in the web page, along with different types of attributes.
- The Inspector also allows you to see the parent-to-child relationship between the individual elements.

PER SCHOLAS

Example: Browser Inspector



perscholas.org
Grow your tech career in only 2-4 months at no cost to you, ever.
APPLY NOW
PER SCHOLAS
MENU
Get the skills you need to grow your career in tech
Jumpstart your tech career with training and job connections at no cost to you!
BECOME A LEARNER
LEARN MORE ›
Elements
Console
Sources
Network
Performance
Memory
Application

```
<!DOCTYPE html>
<html lang="en" js r5s style="--vw:7.85px; --vh:13.19px;" data-whatinput="keyboard"
data-whatintent="mouse" class=" chrome chrome win" initial ready loaded mouse>
 ▸<head>…</head>
…▾<body class="home page-template-default page page-id-1190" data-ol-has-click-
  handler> == $0
  ▸<div id="wisepops-root-407009" class="wisepops-root" style="visibility: visible;"
   data-ol-has-click-handler>…</div>
  ▸<style type="text/css">…</style>
  ▸<noscript>…</noscript>
  ▸<div class="root1">…</div>
  ▸<script>…</script>
  ▸<script type="text/javascript">…</script>
  ▸<script id="leadin-script-loader-js-js-extra">…</script>
   <script src="https://js.hs-scripts.com/8848144.js?integration=WordPress&ver=10.0.
   19" async defer id="hs-script-loader"></script>
   <script src="https://perscholas.org/wp-content/themes/per-scholas/assets/scripts/
   libs.js?ver=1643680791" id="el-extras-js"></script>
  ▸<script id="el-scripts-js-extra">…</script>
   <script src="https://perscholas.org/wp-content/themes/per-scholas/assets/scripts/
   app.js?ver=1675446551" id="el-scripts-js"></script>
   <script defer="defer" src="https://perscholas.org/wp-content/plugins/gravityform
   s/js/jquery.json.min.js?ver=2.6.9" id="gform_json-js"></script>
  ▸<script id="gform_gravityforms-js-extra">…</script>
   <script defer="defer" src="https://perscholas.org/wp-content/plugins/gravityform
   s/js/gravityforms.min.js?ver=2.6.9" id="gform_gravityforms-js"></script>
   <script defer="defer" src="https://perscholas.org/wp-content/plugins/gravityform
   s/js/placeholders.jquery.min.js?ver=2.6.9" id="gform_placeholder-js"></script>
  ▸<script>…</script>
  ▸<script>…</script>
  ▸<script type="text/javascript" id>…</script>
  ▸<noscript>…</noscript>
   <script type="text/javascript" id="hs-script-loader" src="//js.hs-scripts.com/884
   8144.js"></script>
  ▸<script type="text/javascript" id>…</script>
  ▸<noscript>…</noscript>
  ▸<script type="text/javascript" id>…</script>
  ▸<noscript>…</noscript>
  ▸<div style="display: none; visibility: hidden;">…</div>
   <script type="text/javascript" id src="https://assets.adobedtm.com/175f7caa2b90/7
   9023a0eab4c/launch-a45ae8a9c258.min.js"></script>
  ▸<div id="rrTrackPage" style="display: none;">…</div>
  ▸<script type="text/javascript" id>…</script>
   <!--
   Start of global snippet: Please do not remove
   Place this snippet between the <head> and </head> tags on every page of your
   site.
```

PER SCHOLAS

Emmet Toolkit

Emmet is a web-developer’s toolkit that can greatly improve HTML and CSS workflow.

The summary of Emmet below is taken from the Emmet Documentation website:

> “Basically, most text editors out there allow you to store and re-use commonly used code chunks, called *‘snippets’*. While snippets are a good way to boost your productivity, all implementations have common pitfalls: you have to define the snippet first and you can’t extend them in runtime.
>
> Emmet takes the snippets idea to a whole new level: you can type *CSS-like* expressions that can be dynamically parsed, and produce output depending on what you type in the abbreviation. Emmet is developed and optimised for web-developers whose workflow depends on HTML/XML and CSS, but can be used with programming languages too.”

In the next few slides, we will give examples of Emmet syntax and available actions.

PER SCHOLAS

Emmet Toolkit: Syntax

Emmet uses syntax similar to CSS selectors for describing elements attributes and their positions inside of generated trees. Some of the basic operators and their resulting output include:

- The Child Operator, `>`
  - Creates a child element within the current element.
- The Sibling Operator, `+`
  - Creates an element on the same level as the current element.
- The Multiplication Operator, `*`
  - Creates a number of elements based on the given multiplier.
- The ID Operator, `#`
  - Adds an id attribute to the element.
- The Class Operator, `.`
  - Adds class attributes to the element.

For example, this line…

```
div#header+div>ul.class1>li*5
```

… will produce the following code:

```
<div id="header"></div>
<div>
    <ul class="class1">
        <li></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
    </ul>
</div>
```

[Click here for a full reference on Emmet Syntax.]

PER SCHOLAS

Emmet Toolkit: Actions

Emmet also offers a variety of actions to quickly edit code. Some of the available actions, with descriptions taken from the Emmet Actions documentation, include:

- Match Tag Pair
  - Selects content, and/or opening and closing HTML tag name from current caret position (a.k.a “balancing”). Super-awesome implementation that *works even in non-HTML syntaxes*! Implicitly used by many Emmet actions.
- Go to Edit Point
  - Quickly traverse between important HTML code points.
- Toggle Comment
  - Toggles comment. Unlike basic editor’s implementations, matches HTML tag, CSS property or rule when there’s no selection.
- Split/Join Tag
  - Splits (<tag /> → <tag></tag>) or joins (<tag></tag> → <tag />) HTML/XML tag under current caret position.
- Evaluate Math Expression
  - Evaluates simple math expression.
- Reflect CSS Value
  - Automatically copies CSS value under current caret position to all vendor-prefixed variants.

Click here for a full reference on Emmet Actions.

Knowledge Check

- What are the file extensions for HTML files?
- How do you make a comment in HTML?
- What is the name of the file that generally contains homepage content on a website?
- What is the difference between an absolute and relative path?
- What characters and symbols are used to create file paths?
- How do you open the browser inspector on a webpage?
- How can the browser inspector be used to analyze a webpage?
- What is the Emmet toolkit, and why is it useful?
- What are some common operators in the Emmet toolkit, and how are they used?

PER SCHOLAS

Summary

In this lesson, we explored HyperText Markup Language (HTML), which is a markup language used to design web pages. It allows the creation of web content, and tells the browser how to display that content.

HTML is structured using elements, which are objects within the document that are used to organize content:

- Tags are used at the beginning and end of elements to define the element, but void elements do not need end tags.
- Most HTML files should include a **`<!DOCTYPE>`**, **`<html>`**, **`<head>`**, **`<title>`**, and **`<body>`** element.
- Elements can be nested inside of other elements, creating a parent-to-child relationship.
- Elements come in either block-level or inline formats.
- HTML comes with many different elements, each with their own functions and attributes, to help structure content in an organized manner.

Additionally, there are a few tools and techniques available to make content creation more efficient:

- HTML comments can be used to improve code organization and readability.
- Absolute and relative paths are used to define file locations for various purposes.
- The browser inspector can be used to analyze web pages.
- The Emmet Toolkit can be used to speed up development of HTML content.

PER SCHOLAS

Questions?