

Intro to HTML

CSCI 185, Fall 2022

Intro to Computer Programming for the Web

Announcements

1. I changed my office hours! Now M,W,F 2-3PM or by appointment
2. [Tutorial](#) this Friday
3. If you haven't yet completed the [survey](#), please do!
4. Assigned readings:
 - a. [Intro to HTML](#) (for today)
 - b. [History of the web / internet](#) (for Monday)
 - c. Intro to CSS (for Wednesday)

Outline

1. Intro to HTML
2. Rules of thumb
3. Linking to resources
4. Organizing content into containers
5. Activity

Outline

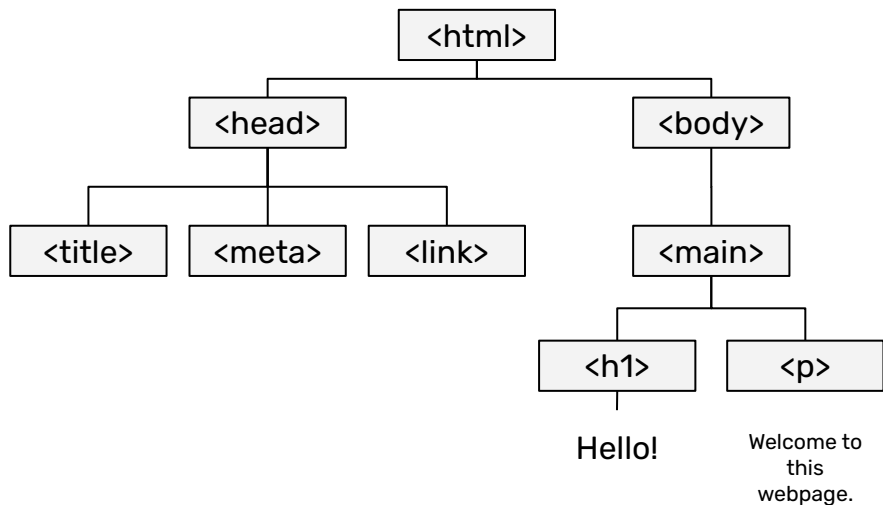
1. **Intro to HTML**
2. Rules of thumb
3. Linking to resources
4. Organizing content into containers
5. Activity

Intro to HTML (Hypertext Markup Language)

HTML is a way of creating web documents using “markup tags”

1. Each HTML tag has a set of rules that you have to follow to correctly use the tag.
2. Sometimes, tags need to be nested in a particular way to be understood by your browser.

How the Browser Interprets HTML



HTML File

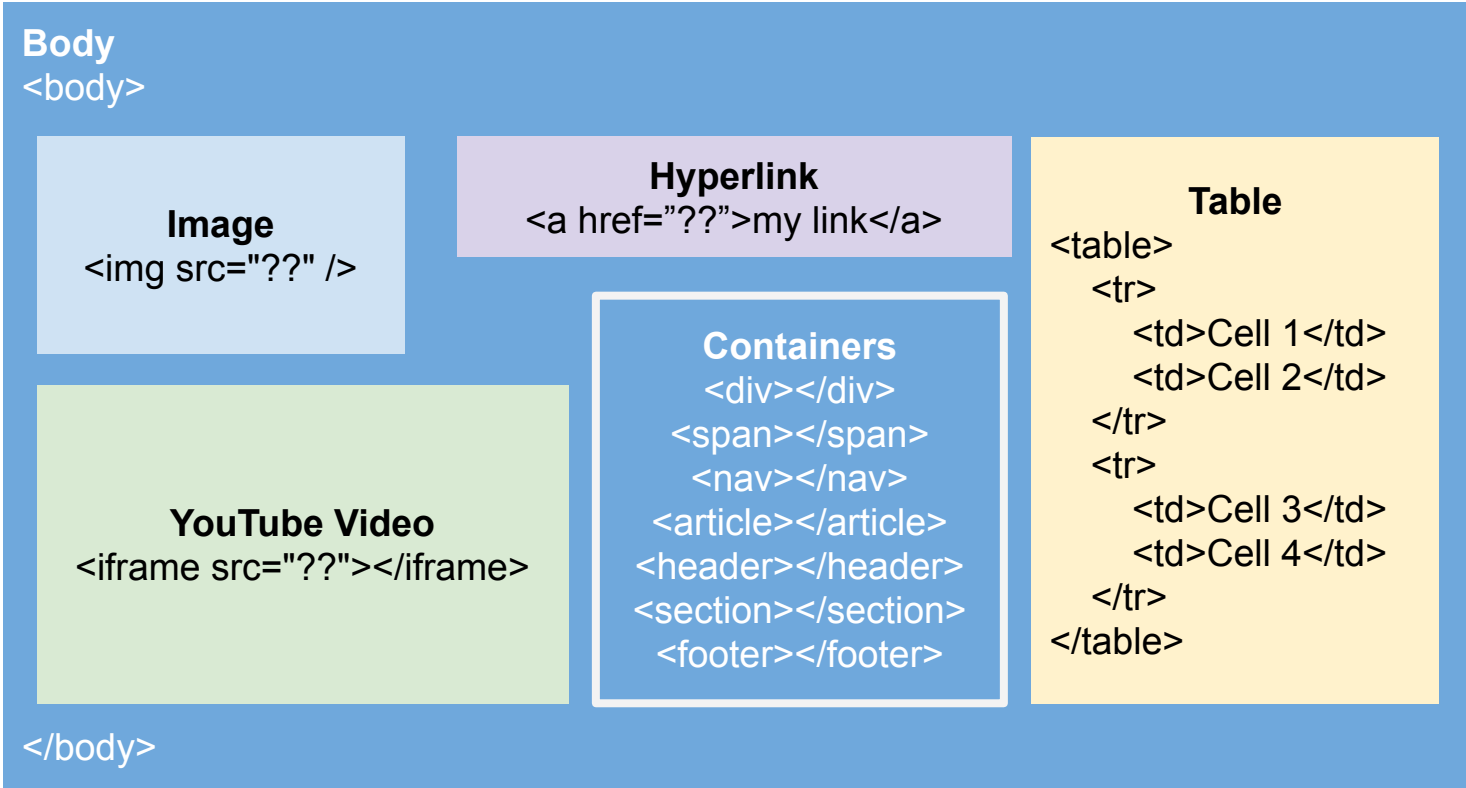
**Invisible section
(for metadata)**

```
<html lang="en">
<head>
  <title>DOM Example</title>
  <meta name="author" content="CSCI 185">
  <link rel="stylesheet" href="main.css">
</head>
```

```
<body>
  <main>
    <h1>Hello!</h1>
    <p>Welcome to this webpage.</p>
  </main>
</body>
</html>
```

**Visible section
(99% of your stuff goes here)**

Lots of elements can go inside of the body element



Outline

1. Intro to HTML
2. **Rules of thumb**
3. Linking to resources
4. Organizing content into containers
5. Activity

1. Avoid spaces, capital letters, and special characters when naming files

When creating new HTML files, it is important to follow the naming conventions listed below:

1. **No whitespace**

Rename `page 1.html` → `page_1.html` or `page1.html`

2. **No capitalization; all lowercase**

Rename `Page1.html` → `page1.html`

3. **No special characters (',*!^%#). Dashes & underscores are OK**

Rename `Jenny's Page!.html` → `jennys_page.html` In addition, all HTML files end with either the `.htm` or `.html` file extension.

2. Most tags have an opening tag and a closing tag

<h1>My Heading**</h1>**

But some don't:

1. Images: ****
2. Line Breaks: **
**
3. Horizontal Rules: **<hr />**
4. Stylesheet Links: **<link rel="stylesheet" href="my_style.css" />**

You'll eventually figure out the rules as you continue building web pages. You can also consult the [HTML Reference](#) to learn more about the rules of each individual tag.

3. The browser ignores whitespace

The browser ignores whitespace:

<h1>My Title**</h1>**

...is interpreted the same way as...

<h1> My
Title
</h1>

4. Make your code readable by indenting and using line breaks

Make your code readable by indenting and using line breaks. Please don't do this:

```
<main><p>Welcome, <strong>Leonard</strong></p><ol><li>item  
1</li><li>item2</li><li>item 3</li>  
</ol></main>
```

4. Make your code readable by indenting and using line breaks

Instead, do this:

```
<main>
  <p>
    Welcome, <strong>Leonard</strong>
  </p>
  <ol>
    <li>item 1</li>
    <li>item 2</li>
    <li>item 3</li>
  </ol>
</main>
```

5. Attribute syntax

Many tags have required or optional attributes (e.g. a tags, img tags, input tags, etc).

Ensure that your attributes are always followed by an equals sign and values are surrounded by quotation marks.

Example:

```

```

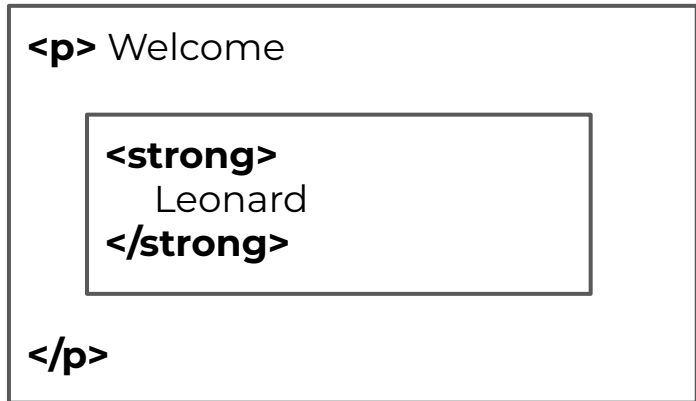


No space between attribute, equals sign, and quotations

6. Last in, first out (LIFO)

Correct

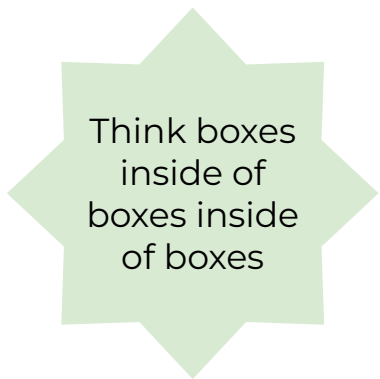
```
<p>Welcome,  
  <strong>Leonard</strong>  
</p>
```



```
<p> Welcome  
  <strong>  
    Leonard  
  </strong>  
</p>
```

Incorrect

```
<p>Welcome,  
  <strong>Leonard</p>  
</strong>
```



Think boxes
inside of
boxes inside
of boxes

7. Use comments to help you understand your code

```
<!-- Welcome Section -->
```

```
<section>
```

```
  <p>
```

```
    Welcome, <strong>Leonard</strong>
```

```
  </p>
```

```
  <ol>
```

```
    <li>item 1</li>
```

```
    <li>item 2</li>
```

```
  </ol>
```

```
</section>
```


Outline

1. Intro to HTML
2. Rules of thumb
3. **Linking to resources**
4. Organizing content into containers
5. Activity

Linking to Resources

Linking is perhaps the biggest idea of the web: documents link together creating a “web” of networked resources.

Many different HTML tags use the concept of linking:

1. Stylesheet references
2. JavaScript references
3. Multimedia embedding (e.g., images, videos, audio files)
4. Hyperlinks

Linking to Resources

Absolute links

- When the file isn't on your computer, you have to specify the server name, and then the path to the file.
- Example: <https://i.pinimg.com/originals/ac/f4/9b/acf49bd0f42b441160a9363dce88b243.jpg>

Relative links

- When the file is on your computer, you specify the file path **relative to your current file**.
- Example: `../images/my_puppy.jpg`
Go up one directory, then into the images directory, and then access the "my_puppy.jpg" image.

Internal links

- When you want to jump to a spot on your current page.
- Example: `#contacts`

Outline

1. Intro to HTML
2. Rules of thumb
3. Linking to resources
4. **Organizing content into containers**
5. Activity

There are lots of different containers that you can use to organize your content...

Before HTML5 (2014)

<code><div></div></code>	<code><!-- block-level container --></code>
<code></code>	<code><!-- inline container --></code>

Current: Semantic Tags (there are more...just a sample)

<code><nav></nav></code>	<code><!-- block-level container --></code>
<code><article></article></code>	<code><!-- block-level container --></code>
<code><header></header></code>	<code><!-- block-level container --></code>
<code><section></section></code>	<code><!-- block-level container --></code>
<code><main></main></code>	<code><!-- block-level container --></code>
<code><footer></footer></code>	<code><!-- block-level container --></code>

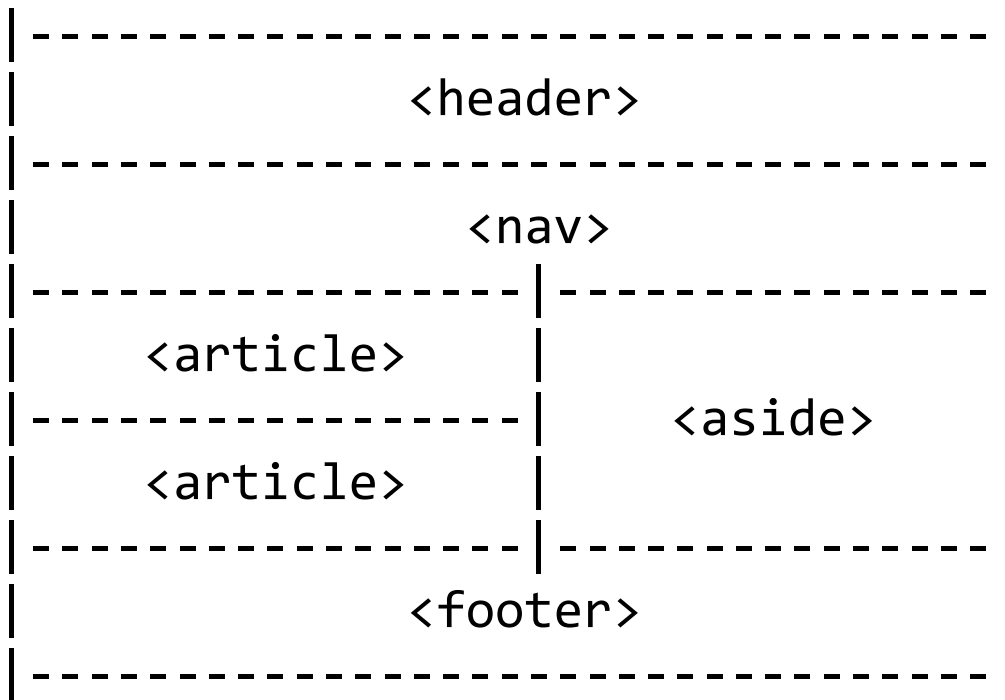
Containers Before (don't do this)

<code><div id="header"></code>	
<code><div id="nav"></code>	
<code><div id="news"></code>	
	<code><div></code>
<code><div id="links"></code>	
<code><div id="footer"></code>	

Not very descriptive of what each section means.

- Harder for web crawlers to effectively index your site.
- Harder for screen readers
- No context cues

Containers After (do this)



Descriptive, semantic indications of what each section means.

- Easier for crawlers and screen readers to target sections of your page
- Easier to read your code.

Outline

1. Intro to HTML
2. Rules of thumb
3. Linking to resources
4. Organizing content into containers
5. **Activity**

Let's look at some examples

Let's download and organize today's files

- On your Desktop (or wherever you typically store your work), create a **csci185** folder. If you're using the lab computers, first create a folder that has your name, and then inside of that folder, create a **csci185** folder.
- Inside **csci185**, create another folder called **lectures**.
- Please download and unzip the [exercise files](#) (lecture02), and save the unzipped folder (**lecture02**) inside of your **lectures** folder.

Let's open your files using Visual Studio Code

- Open VS Code
- Go to File > Add folder to workspace...
- Navigate to the lecture02 file you just created and highlight it.
- Click “Add”

You should now see two folders (**bakery-example** and **exercise**) in the left-hand panel.

Let's preview your files in the Web Browser

- Navigate to **lecture02** using the File Explorer (Windows) or Finder (Mac).
- Open the **bakery-example** folder
- Double-click the **index.html** file.
 - The double-click is a way of asking your browser to read and interpret the index.html file you just made.

Code Walkthrough of Bakery Example

Two things to notice:

1. The structure of the containers
2. How relative linking works

In Class Activity

Practice with:

1. Text markup
2. Containers & semantic containers
3. Media
4. Compound tags

If you don't have a code editor installed yet: <https://codepen.io/vanwars/pen/xxpLzEO>

Specific Tasks (inside of the **exercise** folder)

1. Pick a theme (favorite artist, “about me,” places you’ve travelled, etc.)
2. Add a heading and a paragraph element ([hint](#))
3. Add 3 images. Two should link to images in the images folder, and one should link to an image on the internet ([hint](#))
4. Embed a YouTube or Vimeo video: ([hint](#))
5. Add a list ([hint](#))
6. Add a table ([hint](#))
7. Add some semantic tags to give your sections meaning ([hint](#))
8. If time, try to change the background color (hint: look at the stylesheet).