HTML

WHY do we learn HTML as testers?

- To understand why HTML, CSS, JavaScript is used.
- To be able to read and understand high level HTML structure.

- Front-end, UI (User-interface), GUI (Graphic User-Interface)

- Everything we see and interact with on the web page

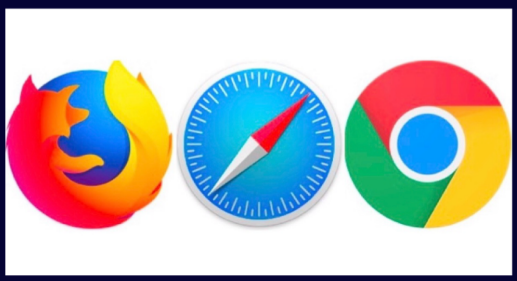
- Backend:

- Everything else other than the front-end is "backend"

- Backend is what constructs the data and delivers it back to users

- Backend does the calculations, creates the business logic, and makes sure user are displayed with correct information

What is the purpose of the web browsers?



- The only purpose of the web browsers is to interpret/translate your files that should be displaying your web site.
- So basically it translates and displays the HTML+CSS+JS in a way that people can understand and use.

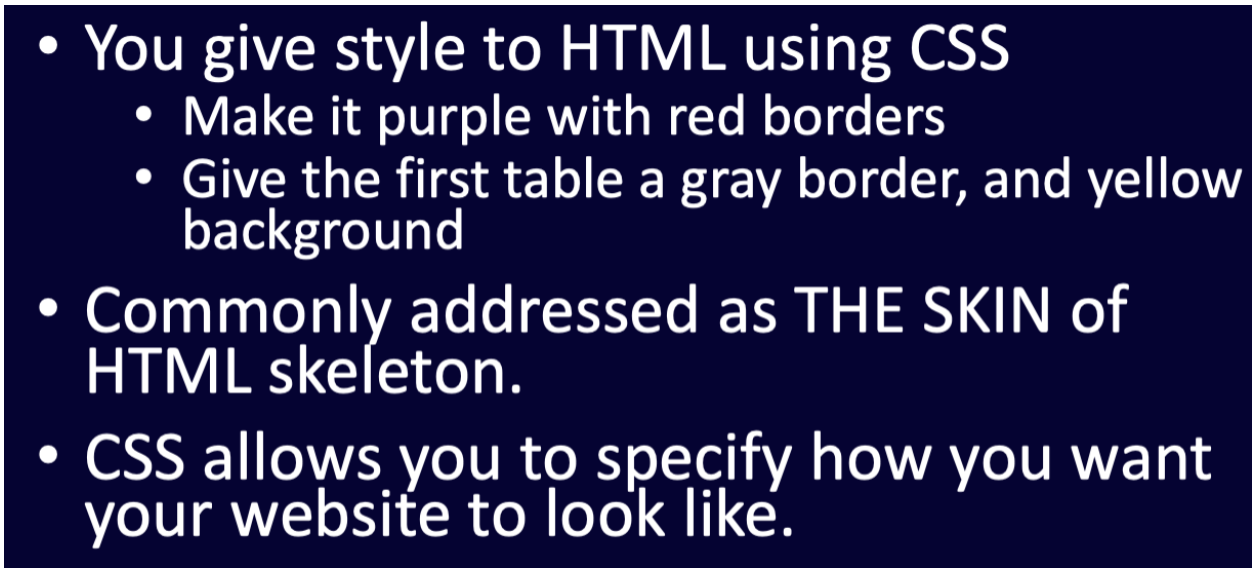
- HTML:

- HyperText Markup Language

- It is a markup language, **not a programming language.**

- HTML does **not have any programming logic** such as variables, loops, or conditions.

- HTML is **used to display content** in a structured manner. HTML codes are **NOT compiled, ONLY interpreted** by the browsers.



- CSS:

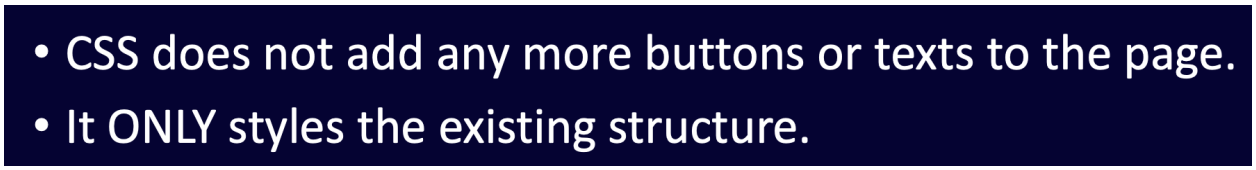
- Cascading Style Sheets

- It is **used to style EXISTING** HTML code

- CSS can **NOT be used alone**

- You cannot create elements using CSS.

- You can **only give style** to existing elements.



- JavaScript:

- JS is a scripting language.

- JS is used to **give actual dynamism, functionality** to the web page.

- car & carpet

- apple & pineapple

- egg & eggplant

- **java** & **java**script / So **zero connection between java**

- What is a web element?

- **Everything we see on a web page** is a web element

- Such as: buttons, links, paragraph, header, images

- What is a TAG?

- Tags act like **containers**.

    - It will **change the display or behaviour of the content** that we passed into it.

- Depending on which tag we choose, we can select the way the content will be displayed for the user.

- img

- p

- How many different types of tags do we have?

- We have 2 types of tags.

#1- Paired tag:

- Paired tags **have 2 parts**

- <**opening tag>** and </**closing tag**>

- the actual content goes in between the opening tag and closing tag

#2- Un-paired tag:

- These are **self-closing tags.**

- They are **just 1 part.**

- All of the information goes into the tag itself.

</selfClosing> such as **<b> , <input>**

Review:

- What is the front-end?

- What user **sees and interacts with** is called front-end

- UI = user interface

- GUI = graphic user interface

- What is the backend?

- Back-end is **everything other than the front-end**

- Back-end **constructs the business logic, the calculations for the front-end**

- Without proper back-end connections, the front-end functionalities will not work properly

- What is HTML?

- Hypertext Markup Language

- What is HTML used for?

- HTML is **used to create certain structure (skeleton)** for web applications

- What is a markup language?

- Markup languages are created **using "mark-up tags"**

- Is HTML the only markup language?

- No. There are other markup languages **such as XML.**

- What is a MARKUP TAG?

- Markup tags act like **containers**

- These **tags will determine how the content will be displayed** on the web page

- We can have the exactly **same content in different tags, and they will be displayed/structured differently** on the web page

- How many types of markup tags do we have?

- 2 types.

#1- Paired tags :

- Paired tags have an opening tag and closing tag.

syntax: <openingTag> CONTENT </closingTag>

- ex: p, h, li, ul, ol, strong, em, u, html, head, body, title, tr, td

#2- Unpaired tags :

- These are also called self-closing tags.

- They are just one part.

syntax: </selfClosing>

- ex: br, img, hr

- What kind of structure does EVERY HTML page have to follow?

#1- Doctype declaration / MUST

#2- <html> </html> tag : / OBLIGATORY

- is the parent/ascendent of all other web elements

#3- <head> </head> tag : / OBLIGATORY

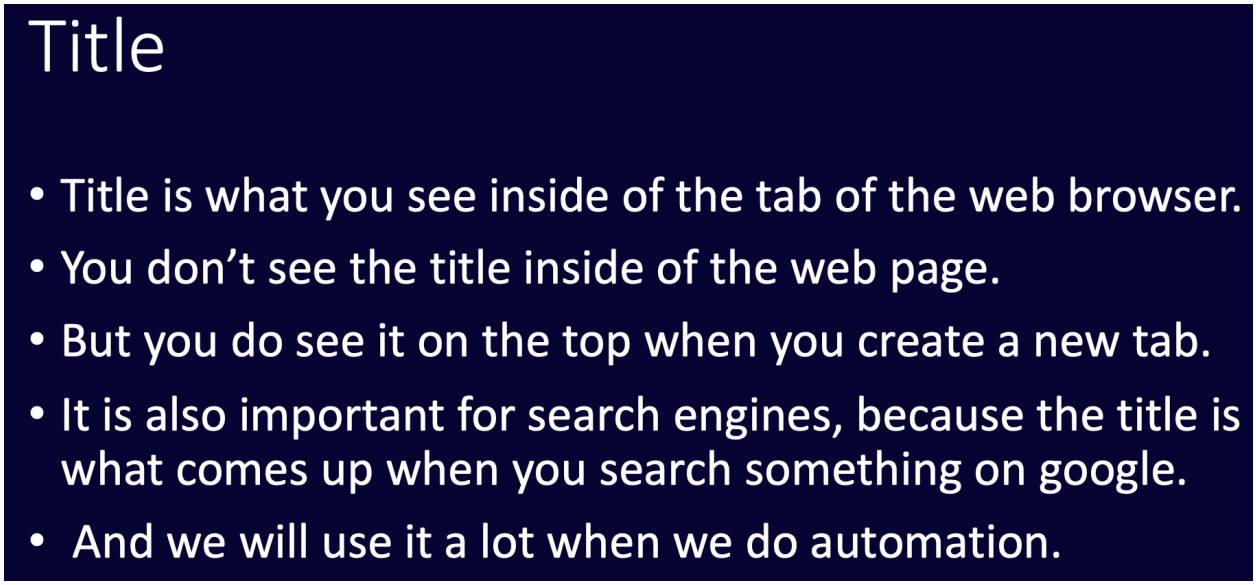
- Everything that is not displayed on the page itself goes inside of the <html> tag, such as: title, links, fontstyle

#4- <body> </body> tag: / MUST

- Everything we want to display on the page goes inside of the <body> tag

–> What is <title> tag, and why is it used for?

- Title of the page is what **comes up in the search engines**
- Title also **defines different pages** of the application
- If title changes it means we are on a different page
- Title is **only displayed on the browser** tab itself
- We **don't see the title in the page itself**



TAGS

–> <p> tag:

- Whatever content is passed inside of this tag, it will be displayed as paragraph

- p tag is a **block element**?

- What is a block element?

- Block element means the element will **take the whole line from left to right side** of the screen

- Anything **comes right after it will be pushed to under** the element

---

-> <h>

- h tag is used for creating **headers**

- h1 thru **h6** will create different size of headers (**h7** does not exist)

- **h1 biggest**, h6 smallest

- ANCHOR TAG : <a>

- <a> tag allows us to **create links** on html page

- <a> tag is a **paired tag.**

- <a> tag **MUST have an href attribute** within.

- The text passed inside of the href attribute's value will be where the user taken once the link is clicked.

- The UI will display the text of the anchor tag <a>, which is in between the opening tag and closing tag.

ex:

** KBB **

- "KBB" text will be displayed on the page

- "href" value (https://www.kbb.com) is where the user is taken when the link is clicked

---

- <table> tag

- This tag allows us to create HTML tables on the page

- By itself it is not enough, we need to use <tr>, and <td> (<th>)

- HTML tables are created **first ROW BY ROW, then cells are created within rows.**

- Tables also should have a head and body section.

<table> </table> : creates the table

<thead> </thead> : contains header information

<tbody> </tbody> : actual content of the table goes here

<tr> :

- stands for **"table row"**

- creates each row

- however many tr we create, that many rows we will have in the table

<td> :

- stands for **"table data"**

- allows us to create cells within the rows

<th> :

- stands for **"table header"**

- allows us to crate cells just like <td> but it will also make the content "bolded" and "centered"

**- **

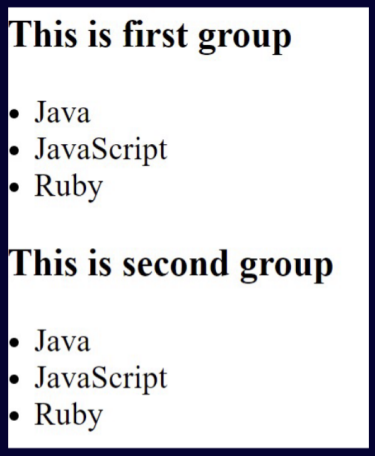
- div tag is commonly used as a **container to style group of web elements**

- it is a **block element** which means it goes all the way to right

- it is just a container to group and apply different stylings, fonts, colors to web elements.

The element allows you to group a set of elements together in one block-level box.

```
<body>
    <font size="10">
    <div>
        <h3>This is first group</h3>
        <ul><li>Java</li>
            <li>JavaScript</li>
            <li>Ruby</li></ul>
    </div>
    <div>
        <h3>This is second group</h3>
        <ul><li>Java</li>
            <li>JavaScript</li>
            <li>Ruby</li></ul>
    </div>
    </font>
</body>
```



**- **

- span is very **similar to div.**

- it is used as smaller container to give certain **styling to parts of the web element without disrupting the rest** of the web element

- span is **inline element (not block element)**

The element acts like an inline equivalent of the element. It is used to either:

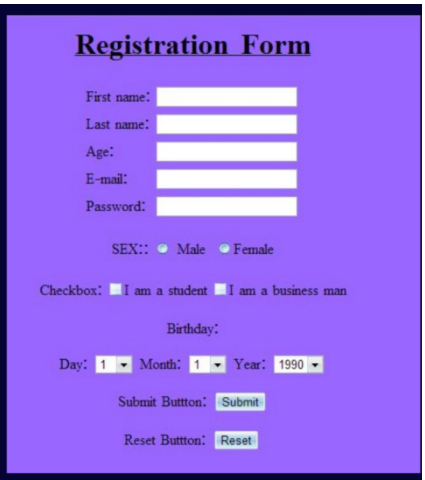
1. Contain a section of text where there is no other suitable element to differentiate it from its surrounding text.
2. Contain a number of inline elements.

```
<body>
    <h1>This is <span style="color:red">how</span> <span style="color:green">span
    works</span></h1>
</body>
```

- input tag

- <input> tag allows the user to **enter input in different ways**
- username
- password
- checkbox
- radiobutton
- colorpicker
- calendar

- Forms are basically a container for all of these different types of **inputs**.
- Forms are created using <form></form> tag.
- Checkboxes, buttons, dropdown menus, and color pickers go in form tag.
- Forms are sending data to server or database or wherever we set it up to sent.
- The form we create could be just creating a **get** request.
  - Example: Google search, or logins
- Or it could be creating a **post** request;
  - Example: Signing up to a website



- select tag

- <select> tag allows us to **create dropdowns** in HTML pages.
- it is a **paired tag**
- we must provide <**option**> tag inside <**select**> tag to provide multiple options for the user

<select>

<option> January </option>

<option> February </option>

<option> March </option>

</select>

-> What is an ATTRIBUTE?

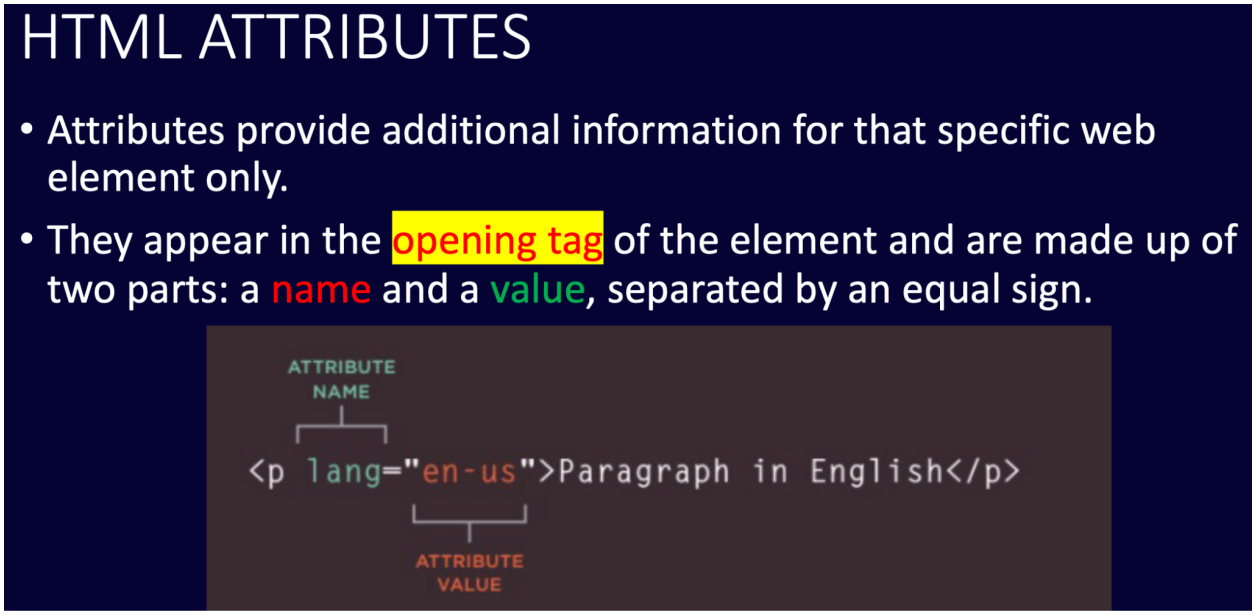
- Attributes provide **additional information** about a given tag/specific web element.

- We can have as many attributes as we want to have.

- Attributes will **always go inside of the opening tag** (if it is a paired tag)

- If it is an unpaired tag (self-closing tag), it will go inside of the tag itself.

syntax:



<openingTag **attributeName1="attributeValue"** attributeName2="attributeValue">
CONTENT </closingTag>

</selfClosingTag attributeName1="attributeValue">

**ex: today is a snowy day **

#1- What is the **tag**?

-

#2- What is the text/content of this tag?

- today is a snowy day

#3- What is the **name** of the attribute used?

- style

#4- What is the value of the attribute used?

- color : red;

SHORTCUTS

---> To **copy the path** of a file:

MAC:

#1- right click on the file

#2- press and **hold option** button from keyboard

#3- select "**copy as path name**"

#4- paste the copied text in the img src

WINDOWS:

#1- right click on the file

#2- go to properties

#3- go to security tab from above

#4- get the path from there

How to get path in different OS (MAC & WINDOWS):

- If the file that we are trying to get the path of is inside of the same folder with the source, **just the name and the extension** of the file is good enough to pass as path.

ex: sunnyday.jpeg is in the same file as our firstTask.html

- Therefore just the name of the file is good enough as a path.

<img src="sunnyday.jpeg" height="300">

- If they are not in the same file, we have to pass the full path (absolute path).

How do we get the **absolute path (full path):**

- MAC:

  First way:

  #1- Right click on the file

  #2- Click **"Get Info"**

  #3- Copy from "Where" section: /Users/cybertekchicago-1/Desktop/HTML Class

  Second way:

  #1- Right click on the file

  #2- Press and hold "option" button from keyboard

  #3- Select "Copy as path name"

  #4- Paste it wherever we need to use it.

- WINDOWS:

  First way:

  #1- Right click on the file

  #2- Go to "Properties"

  #3- Go to "Security" tab from the top of the page

  #4- Copy path from there and paste where needed

  Second way:

  #1- Right click on the file

#2- Go to "Properties"

#3- Copy from the "Location" and paste in where ever needed