A. HTML
    a. Hyper Text Markup Language
    b. Purpose: to give structural meaning to web content
    c. https://html.spec.whatwg.org
    d. Derived from a larger family of markup languages called SGML (standard generalized markup language)
        i. Tim Berners-Lee, inventor of HTML, intended it to be an application of SGML
        ii. The design was inspired by SGML tagging,
        iii. no clear expansion and parsing guidelines were established, so most HTML documents are not valid SGML documents
    e. HTML5 is the most recent version (2008)
        i. Added several semantic tags
        ii. Added more robust support for
            1. video + audio
            2. vector graphics (svg, canvas)
            3. web workers (js running in background)
    f. Viewing HTML in browser
        i. Demo in Chrome
        ii. Use browser comfortable and familiar to you
        iii. Things may look different
    g. Open file in browser
        i. Menu command: File > Open File
        ii. Navigate to file location
B. Semantic HTML
    a. https://internetingishard.com/html-and-css/semantic-html/
    b. Use of HTML markup to express the meaning of the content instead of defining presentation
    c. Separation of concerns
        i. HTML: markup, content structure
        ii. CSS: presentation, appearance
        iii. JS: interaction
    d. Example:
        i. <H1> vs big & bold
    e. Important to make structure semantic
        i. Maintainability: helps you as a developer keep your site organized
        ii. Accessibility
            1. Every HTML document has an “outline,” which is how search engines and screen readers view the hierarchy of the content on the page
            2. Outline helps adapt the way the browser presents information to the users according to the structure of the document
            3. The more semantic the markup, the easier it is for search engines, screen readers, and other machines to identify the different parts of your website.
        iii. Picture a series of boxes tucked away in an attic
            1. None of the boxes are labeled
            2. How do we know how to organize whatever is inside the boxes when we visit the attic?
                a. We could unpack each box every time we go up there
                b. Uses up a lot of time, energy

3. Semantic HTML = giving the boxes relevant labels to give structure/meaning to whoever has to view the content later
    a. User's browser
    b. Web crawler/robot
    c. Code maintainers

C. What does HTML look like?
  a. Elements reference: https://developer.mozilla.org/en-US/docs/Web/HTML/Element
  b. Content surrounded by (contained by) tags
    i. Tag = some keyword between < > brackets i. <tag>
      1.
      2.
      3. <img>
    ii. Open tag + (USUALLY) closing tag
      1. <tag></tag>
      2.
    iii. Closing tags enclose content encompassed by the tag
      1. Foo
      2. Bar
      3. <img /> ← images don't have content to enclose
    iv. "Self closing tags"
      1. Closing tag is optional
        a. it's implied that a new tag would not be able to be started without closing it
        b. html, head, body, p, dt, dd, li, option, thead, th, tbody, tr, td, tfoot, colgroup
      2. Tags that never take an explicit closing tag, can use />
        a. img, input, br, hr, meta
      3. If unsure, use the HTML validator: https://validator.w3.org/
  c. Block vs. inline elements
    i. Block
      1. block-level elements may contain inline elements or other block-level elements
      2. block elements create "larger" structures than inline elements
    ii. Inline
      1. inline elements may contain only data and other inline elements
      2. We can't put block elements inside inline elements, invalid HTML
      3. inline elements do not force a new line to begin in the document flow
  d. HTML can be nested
    i. Examples
      1. Inline elements are nested within block elements
        a. I'm nested!
      2. Block elements can be nested within block elements
        a. <section> I'm a paragraph I'm another paragraph </section>
      3. Inline elements can be nested within inline elements, in some cases
        a. I'm emphasized but I'm not
    ii. Nesting must be closed from inside out
      1. Cannot cross tags while closing nesting
      2. Example:
        a. Correct: <section> This is emphasized </section>

b. Wrong: <section> This is emphasized</section>

D. HTML tags
   a. Tag name and attribute names for HTML elements may be written with any mix of lowercase and uppercase letters; they are case-insensitive.
      i. The recommendation is that you write them in lowercase for consistency and easier readability
   b. <!doctype html>
      i. Informs the website visitor's browser that the document being rendered is an HTML document
      ii. It tells how the document should be interpreted, by indicating what version or standard of HTML is being used
      iii. Not actually an element or HTML tag itself
      iv. Every HTML5 document (ie, all new web documents) should begin w/ DOCTYPE declaration to be compliant with HTML standards
      v. It should always be the first element in the document
      vi. It has no closing tag and is NOT self closing

E. <html> tag
   a. the root (top-level element) of an HTML document
   b. Also called “the root element”
   c. All elements must be descendants of this element, except <!DOCTYPE html>
   d. lang attribute
      i. Define the language of an element
         1. If the element is uneditable by user, lang indicates the language the element content is written in
         2. If the element is editable by user (form inputs, eg), lang indicates the language user should use to enter data
      ii. Could tag/define every single element of a page as different language
         1. For example, if you had quotations from many different sources/languages, you might want to tag each with a lang attribute to provide language context
         2. In all cases, we should at least define it on HTML tag to apply to the entire document as a default
            a. Define it on the HTML tag so that it applies to the content in the HEAD as well as the BODY
   e. dir attribute
      i. Indicates the “directionality” of language, ie, what direction you read the text
         1. Ltr = left to right, eg. English, Spanish
         2. Rtl = right to left, eg. Hebrew, Arabic
         3. Auto = let browser decide
      ii. Can be overridden by css
         1. It is recommended that you define directionality via HTML attributes for cases where CSS not supported for some reason

F. <head> tag
   a. provides general meta-information about the document itself
   b. provides an area to link to scripts and style sheets

G. head tag children
   a. <meta> tags
      i. https://developer.mozilla.org/en-US/docs/Web/HTML/Element/meta
      ii. represents metadata that cannot be represented by other elements

b. Meta tag for character encoding
    i. <meta charset="utf-8" />
    ii. 1st child of head tag
    iii. charset attribute
        1. declares the page's character encoding
        2. You should always specify encoding, because it is needed by the browser to process non-ASCII characters entered by the user in forms, in URLs generated by scripts, etc
        3. Utf-8
            a. A Unicode-based encoding such as UTF-8 can support many languages and can accommodate pages and forms in any mixture of those languages
            b. Its use also eliminates the need for server-side logic to individually determine the character encoding for each page
            c. Reduces the complexity on a multilingual site or application
            d. Unicode encoding also allows many more languages to be mixed on a single page than any other choice of encoding.
            e. Wide browser support
            f. Wide usage
            g. https://w3techs.com/technologies/details/en-utf8/all/all
                i. UTF-8 is used by 93.5% of all the websites whose character encoding we know

        4. Always use utf-8, not only for serving HTML, but for authoring as well!
            a. Just declaring an encoding inside a document or on the server won't actually make your HTML coded that way; you need to save the text in that encoding to apply it to your content
            b. Declaring it in your HTML document helps the browser interpret the sequences of bytes in which the text is stored
            c. set up UTF-8 as the default for new documents in your editor
c. <title> tag
    i. Required child of head tag
    ii. Indicates the content that will be displayed in the browser tab
    iii. title only allows characters – no markup

H. <body> tag
    a. content section of webpage
    b. 1 per HTML document
    c. All visible content in the viewport will be located inside the body

I. Content sectioning
    a. Outliner: https://gsnedders.html5.org/outliner/
    b. HTML5 was a major leap in HTML in terms of bringing precision to how documents are broken into sections using what are called sectioning blocks and headers
        i. More precise tags make document outlines more predictable
        ii. Document outlines generated from the HTML are used by the browser to improve the user experience
    c. HTML content sections
        i. https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Using_HTML_sections_and_outlines

ii. All content lying inside body tag is part of a section
iii. The section may be the body tag itself
iv. sections in HTML5 can be nested
v. Explicit sections means enclosing content in opening/closing tags like header, article, aside
vi. Implicit sections means dividing content with h1-h6 headers and letting the browser figure out the outline
1. Each header causes browser to close previous section and start new
vii. To make your markup human-understandable, good practice is to use explicit tags for opening and closing sections

J. Content dividers that ARE sectioning blocks
a. The following blocks produce new sections in the outline of the document
i. Inherently describe the content contained
b. <header> tag
i. defines an area of the page which typically contains the logo, title, and navigation
ii. header tag can also be used in other semantic elements such as article or section to define the section's heading, author name, etc
iii. Despite its name, it is not necessarily at the beginning of the page or section!
c. <footer> tag
i. defines an area of the page which typically contains the copyright, legal notices and sometimes some links
ii. footer tag can also be used in other semantic elements such as article or section, containing perhaps the section's publication date, license information, etc.
iii. Despite its name, it is not necessarily positioned at the end of the page or section!
d. <main> tag
i. dominant content of the body of a document
ii. Everything that is not a header or footer should probably go in main
e. <article> tag
i. self-contained composition in a document which is intended to be independently distributable or reusable
ii. Examples
1. forum post
2. magazine or newspaper article
3. blog entry
4. Twitter post
f. <nav> tag
i. Indicates a block of navigation links, either to within the current document or to other documents
ii. Examples
1. Menus
2. tables of contents
3. Indexes
4. Breadcrumbs
iii. nav tag can be its own block-level element or be nested within header, footer, etc.
g. <aside> tag
i. portion of a document whose content is only indirectly related to the document's main content
ii. Examples

1. Sidebars
2. call-out boxes
3. Could be used to markup ad space/promoted content/affiliate info

iii. Aside does not imply “to the side” (ie, left or right)

1. Can be located anywhere within content.

h. <section> tag

i. Defines a section of a document to indicate a related grouping of semantic meaning

ii. A section must have a header to be valid

K. <h1>, <h2>, <h3>, <h4>, <h5>, <h6> tags

a. Header for block of content

i. Increasing header numbering as moving lower down into outline

ii. Start with h1, next most important is h2, etc.

iii. Try not to skip headers

b. H1 is the header for the highest level content

i. 1 h1 per page, ideally the most important piece of information

1. Article title
2. Company name
3. Person’s name on a biography page

c. Use in conjunction with hgroup if you want to group more than one together

i. Example: title and subtitle for article

L. Content dividers that are NOT sectioning blocks

a. The following blocks do not produce new sections in the outline of the document

i. They do not inherently contain any meaning about their content

b. tag

i. Block-level content element

ii. Generic box

iii. hook for css

c. <p> tag

i. paragraph of text

d. tag

i. Inline-level content element

ii. Generic box

iii. Like div, mostly used for css hooks

e. <blockquote> tag

i. Long quotation

ii. Usually rendered indented visually

iii. cite attribute

1. Used to provide URL reference to where the quote comes from

f. Lists

i. There are 3 types of HTML lists

1. tag
    a. Unordered lists
    b. Lists that do not have inherent order
        i. Bulleted list of HTML lists
        ii. Todo items that don’t need to be done in a specific order
2. tag
    a. Ordered lists
    b. Lists that have inherent order

i. series of steps
3. <dl> tag
a. list of related term & definition pairs
i. implement a glossary
ii. display list of key/value pairs metadata
ii. Each ul and ol list item is defined by a li tag
iii. Each dl list item pair is defined by
1. <dt> tag for the term
2. <dd> tag for the definition

g. <figure> tag
i. self-contained content that contains and image, illustration, diagram, code snippet, etc., that is referenced in the main flow of a document
ii. Can be moved around without affecting main flow (ie, it doesn't have to be directly located next to its reference)

h. <figcaption> tag
i. Child of figure tag
ii. Used to indicate caption or legend for its parent figure

i. Tables
i. <table> tag
1. Block-level element used for the presentation of tabular data, ie data that can be represented as a 2 dimensional display of columns and rows
2. Tables should NOT be used for laying out elements just to get them in a grid
3. Should only be used for marking up column/row data
ii. <caption> tag
1. caption/title of a table
2. Always first child of the table tag
iii. <thead> tag
1. set of rows defining the head of a column
iv. <tbody> tag
1. Set of rows defining the body of the table
v. <tfoot> tag
1. set of rows defining the foot of a column
2. summarizing the columns of the table
3. Example: totals in a spreadsheet
vi. <tr> tag
1. row of cells in a table
2. Container for combination of th and td tags
vii. <th> tag
1. Header of a group of table td tags
2. Could be located in a thead tag or could be the start of a tr row
viii. <td> tag
1. cell of a table that contains data
ix. rowspan/colspan attributes
1. Allows a single table cell to span the width or height of more than one cell or column
2. Picture "merge cell" in spreadsheet programs
3. rowspan attribute
a. Allows a single table cell to span the height of more than one cell or row

4. colspan attribute
    a. Allows a single table cell to span the width of more than one cell or column
5. rowspan/colspan are attributes of th and td tags
    a. might be used for a header cell that titles a group of columns or a side-bar that groups rows of entries
    b. The value of either attribute must be a positive integer (a whole number)
    c. specifies the number of columns or rows that the cell fills

j. Forms
    i. <form> tag
        1. document section that contains interactive controls
        2. Forms are used for
            a. submitting information to a web server
            b. capturing/handling interactive actions in a human-usable way
        3. action attribute
            a. URI of a form processing script on a server (ruby, php, perl, etc.) that handles the data
        4. method attribute
            a. HTTP verb method of sending data
            b. https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods
            c. In almost every instance, this would be a POST or a GET
        5. We often do not specify method or action for a React app
            a. We use JS to capture the form information and manually handle it (save it to state, send it to API calls, etc.)
    ii. <fieldset> tag
        1. Used to group multiple related fields/controls/labels within a form
        2. Example: first/middle/last name
    iii. <legend> tag
        1. Child of fieldset tag
        2. Caption/title for content of its parent fieldset

k. <label> tag
    i. title/caption for interactive form element
    ii. Best practice: associate label + input with “for” (or “htmlFor” in React app)
        1. allows screen readers and other non-visual browsers to make link between label and input
        2. allows input to be activated when label is activated, especially on very small inputs (eg, checkbox, radio)
    iii. Best practice: 1 label per input
        1. can have multiple labels
        2. Screen readers can have problems differentiating them

l. <input> tag
    i. https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input
    ii. Creates the interactive control to accept data from the user
    iii. type attribute
        1. Method for specifying which type of widget
        2. These are the most common, several more depending on application you need
        3. Not all supported by all browsers!
        4. type=”button”: button widget with no default behavior

        a. Use for an action where you need to capture javascript action and trigger code
    5. type=”text”: default text inputs
    6. type=”checkbox”: allows multiple values to be selected for an input response
    7. type=”radio”: allows only 1 value to be selected for an input response
    8. type=”submit”: button widget with default behavior of submitting the form
    9. type=”password”: password inputs
    10. type=”number”: text input field that only accepts number input
        a. Mobile browsers will launch number-only keypad
        b. Browser provides automatic validation entered text is a number
        c. set of up and down buttons to step the value up and down
    11. type=”telephone”: text input field that has extra functionality for telephone input
        a. Mobile browsers will launch telephone keypad
        b. makes adding custom validation and handling of phone numbers more
        c. convenient
        d. the input value is NOT automatically validated to a particular format
iv. value attribute
    1. The value currently represented by the input
    2. For text fields, it’s the content of the field that the user entered OR what was supplied initially
    3. For checkboxes/radios, the value is programmed and the user indicates if the checked value is on/off or true/false.
        a. If the value is not otherwise specified, it is the string “on” by default
        b. When a form is submitted, only checkboxes which are currently checked are submitted to the server and the reported value is the value of the value attribute
    4. For submit/button, it’s the content of the button
v. disabled attribute
    1. State where user cannot interact with the control
        a. Not clickable
        b. User cannot activate/input value
vi. readonly attribute
    1. User cannot modify the value of the input
    2. Different than disabled: user can still click on/interact with control and trigger onclick handlers, for example
vii. required attribute
    1. Indicates form is invalid if left empty (form will not submit if user does not provide value)

m. <select> tag
    i. control that provides a menu of options
    ii. multiple attribute
        1. allows multiple options to be selected by cmd/ctrl clicking
n. <optgroup> tag
    i. Child of select tag
    ii. Used to group options in a select
    iii. Creates a label which is not selectable by the users, under which the grouped options are displayed
o. <option> tag

i. Defines items contained in a select or optgroup
ii. Contains the text displayed as the option label
iii. value attribute
    1. value to be sent to the form
iv. selected attribute
    1. indicates default selected option
    2. If none specified, defaults to first in the options list
    3. If multiple attribute is specified on select tag, multiple options can be marked selected

p. <button> tag
i. clickable button element
ii. Can be used either inside OR outside forms
iii. When there is no CSS applied, it appears like an OS button by default
iv. Button tag surrounds the content displayed inside the button borders

M. <image> tag
a. Embeds image into a document
b. src attribute
i. Specifies the path to the actual image file
ii. As with any uri, path could be relative (ie relative to the current file on the same server) or absolute (a URL starting with http)
iii. width & height attributes
    1. We can set these on the tag in html to set up a stable page layout so that when the images and other elements are loaded, the page doesn't shift around distractingly
    2. You can additionally set width and height in CSS (for responsiveness, for example)

c. alt attribute
i. Alternate text to indicate the picture contents for users who cannot experience it visually
ii. Remember week 1 when we talked about including alt text for accessibility
iii. In addition, alt text is displayed when the image cannot be displayed due to a loading error, for example

N. Content/typography markup
a.
 tag
i. Line break
ii. Equivalent to carriage return
iii. Break up lines of text that are still related by block

b. <a> tag
i. creates a hyperlink to some other page or element
ii. tag surrounds text, image, etc. to be clicked on
iii. Examples
    1. Absolute link = Google
    2. Relative link = Foo
    3. Mailto: = April's email
iv. target attribute
    1. _self: load url into current browsing tab or window -- this is the default
    2. _blank: load url into new browsing tab or window

c. , , <b> and <i> tags
i. and

1. These tags have different semantic means, despite seeming confusingly similar
2. Strong
    a. Content with strong importance, including things of great seriousness or urgency
    b. Example: **Warning!** This is **very dangerous**.
3. Em
    a. for words that have a stressed emphasis compared to surrounding text, which is often limited to a word or words of a sentence and affects the meaning of the sentence itself.
    b. Example: I really like carrots: I *love* carrots vs. I like carrots as opposed to peas: I love *carrots*

ii. and tags

1.
    a. used to draw attention to text without indicating that it's more important
    b. Use it only when nothing else is appropriate
    c. Example: The two most popular science courses offered by the school are **chemistry** (the study of chemicals and the composition of substances) and **physics** (the study of the nature and properties of matter and energy)
2.
    a. a range of text that is set off from the normal text for some reason. Some examples include technical terms, foreign language phrases, or fictional character thoughts
    b. Example: The Latin phrase *Veni, vidi, vici* is often mentioned in music, art, and literature. (demonstrates that Veni, vidi, vici is a different language)

d. <sub> and <sup> tags

i. <sub>

1. Subscript
2. Chemical symbols: $C_8H_{10}N_4O_2$

ii. <sup>

1. Superscript
2. Footnote numbers
3. Exponents: a ^ 2 (a^2)
4. Ordinal numbers: 4^{th}

e. <abbr> tag

i. Abbreviation or acronym

ii. title attribute

1. Instructs browser to give definition inline

iii. <abbr title="Northeastern University">NEU</abbr>

f. <address> tag

i. contact information associated with the webpage itself, ie whom you would contact if you wanted to talk with the person/company publishing the information

ii. Is not used for just any random address

iii. Examples

1. providing a business's contact information in the page header
2. indicating the author of an article

g. <q> tag
   i. Short inline quotation
   ii. Browser will render in quotation marks
h. <s> tag
   i. Text that is no longer relevant or accurate but that is retained in the content as a record that something changed
   ii. Example: a todo list where each item is crossed off, but still in the list as an archive
i. <time> tag
   i. presenting dates and times in a machine readable format
   ii. Date/time enclosed by tags can be human readable
   iii. datetime attribute
      1. needs to be valid machine readable format
      2. https://developer.mozilla.org/en-US/docs/Web/HTML/Element/time
   iv. Example:
      1. <time datetime=”2018-11-22”>November 22, 2018</time>
      2. <time datetime=”2018-11-22”>next Tuesday</time>