# COMP 1537

## HTML Basics I

# COMP 1537

- Arron Ferguson
- Email:
  - **arron_ferguson@bcit.ca**
  - **Subject line [COMP1537]**
- Office Hours (Bby):
  - Zoom meetings, book an appointment

# RULES & INFO (1/4)

- Scheduled lab times are mandatory (attendance is strictly enforced)
  - And recorded!
- Lectures will not be recorded (i.e., synchronous deliver)
  - Encouraging you to "be here" as well as be here
  - Take notes, ask questions, take notes, ask questions (important!)
- All assignments will have hard due date/times
  - Late submission = zero grade for that submission
- All material is found on the learning hub
  - Under the lecture, assignments, & examples sections

# RULES & INFO (2/4)

- Contact information:
  - Both your lab instructor as well as your lecturer:
    - Arron Ferguson
    - arron_ferguson@bcit.ca
- Your lab instructor is your first point of contact for questions/office hours, help – in this case, both me
- Submissions of assignments/exams will be through the learning hub

# RULES & INFO (3/4)

- You are encouraged to collaborate by:
  - Helping each other understand material and assignments
  - Discussing requirements and approaches
- **Plagiarism** is not allowed:
  - Exchanging or sharing code snippets/solutions
  - Submitting someone else's work as your own
- Academic Integrity policy
  www.bcit.ca/files/pdf/policies/5104.pdf

# RULES & INFO (4/4)

- What you will learn: fundamentals of creating web apps
  - See course outline for details
- Expectations (from me):
  - You are here to learn (and like to learn)
  - You will show up on time each class
  - You will take notes and participate during lecture time
    - Remove distractions (e.g., FB, messaging, games, etc.)
  - You will work on your assignments on your own
  - You will ask questions
    - Chances are others will have the same questions
  - You want to make this fun – I do! ☺

# EVALUATION CRITERIA

- Your grading scheme:
  1. Assignments 40%
  2. Midterm 20%
  3. Final exam 40%
- To pass the course, average of midterm and final exams must be 50% or higher

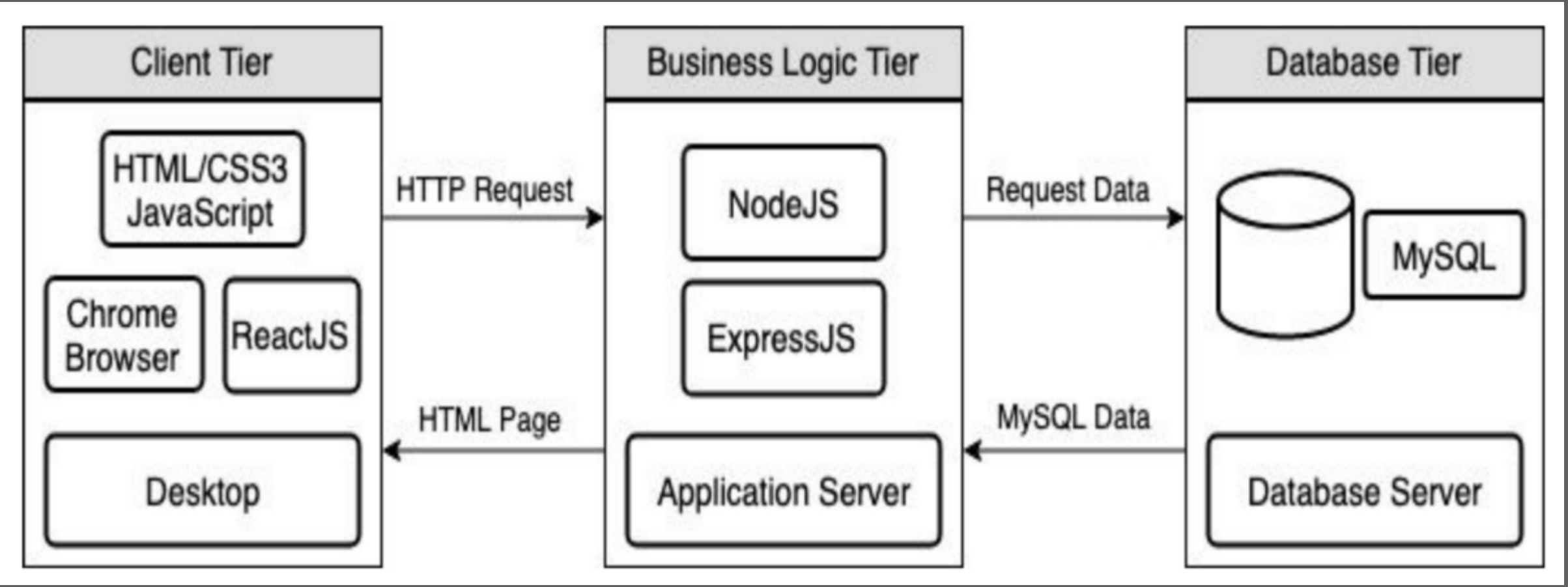
 

# WHAT IS THE WEB

- Web details:
  - Specific protocol used is: HTTP (or HTTPS if the connection is encrypted)
  - Web server receives requests, sends content (HTML, CSS, JavaScript)
    - Web server is a computer that runs web server S/W (e.g., Apache)
    - Can be dedicated or shared (or a network of computers … see cloud later in slides)
    - Usually has plugin modules for running programming languages
      - For dynamic content generation (DCG)
  - Web browser makes requests, accepts content
    - Browser is an application that renders content (e.g., text, images, video, audio)
    - Browsers much handle all kinds of content and types of content
      - E.g., JPEG, PNG, GIF – just for image types alone!

# THREE TIERED WEB ARCHITECTURE



# THE CHANGING WEB (1/2)

- The mobile platform is the *default* platform
  - Mobile == smart phones, tablets
  - Web technologies have recognized this and adapted to it
    - We are in another sudden moment of technology change – similar to 1995-2000
- Cloud computing is the norm
  - e.g., Instagram, DropBox, Box, Amazon EC2, Microsoft Azure, FireBase, etc.
  - Software as a Service (SaaS) – Office 365, Adobe Suite, etc.
- Privacy issues are prominent – and continually being addressed
- Legal/ethical/copyright issues have become prominent
  - E.g., who owns the rights to a YouTube video of video game gameplay?
  - E.g., who's responsible for dating app users posting adverts for illegal activity?

# THE CHANGING WEB (2/2)

- Social Media
  - Another norm we all expect
  - Changing view of computing as too *technical* for the masses
  - Corporations, organizations, governments, all recognize the need to tie in with social media
- The Internet of Things (IoT):
  - The connection of all kinds of devices to the Internet
    - Cars, refrigerators, toasters, home climate control systems/alarms, everything
  - Utilizing five forces: mobile platform, sensors, geo-location, big data, social media (see the book "Age of Context")

# HTML – THE COMMON WEB DIALECT

- Web browsers can play/display
  - Text, images, audio, video as we said
- And so they use a structured markup language known as
  - HyperText Markup Language (HTML)
- HTML has gone through many changes
  - HTML 2.0 in 1995, HTML 3.2 in January, 1997 (handed over to W3C)
  - HTML 4.0 in December 1997 (variants Strict, Transitional, Frameset)
  - HTML 4.01 in December 1999
  - HTML 5 in 2008 as a working draft
  - HTML 5 in 2011 formalizing, and 2014 being a finished recommendation
  - HTML 6 … still in the works!

# ACCOMPANYING WEB LANGUAGES

- There are **two** helper languages that are used with HTML:
  - Cascading Style Sheets (CSS)
    - Used to create styling effects on elements, create rules for layouts, how to embed fonts
      - E.g., make all paragraph fonts purple, ensure that a layout component only shows up if the width of the browser window is larger than 600 pixels
  - JavaScript
    - Program code for the browser window
    - Allows for a wide range of functionality
      - Connecting to a web server, programmatically adding/editing/deleting HTML, CSS
      - Drawing graphics, handling user interaction, validating user input, etc.

# HTML SYNTAX (1/5)

- HTML uses tags
  - Beginning tag, end tag … and stuff in the middle

```
<p>This is a paragraph.</p>
```

  - Tags use the less than, greater than, and forward slash characters
- A begin tag and end tag together form an element
  - Elements must have a begin tag and end tag
    - Unless they use the empty element shorthand
      - E.g., <br/>
    - Empty elements are acceptable if there is no content inside
      - but they are limited to "void elements" (e.g., `br`, `input`, `link`, `meta`, `img`, etc.)

# HTML SYNTAX (2/5)

- Elements can contain other elements too!
  - Which contain other text and elements,
    - Which contain other text and elements …
- Ergo, HTML is hierarchical
- Some elements allow:
  - Only child elements – we call this an element content model
  - No content at all – we call this an empty content model
  - A mix of elements and attributes – we call this a mixed content model

```
<div>
  <p>A <i>very</i> short <b>sentence</b></p>
</div>
```

# HTML SYNTAX (3/5)

- Elements can have attributes

```
<p id="a123">another paragraph.</p>
```

  - Can have more than one attribute (even many)
  - Attributes do not have any implied order
  - Attribute name is followed by the equals sign, begin quotes, the value, end quotes
  - Attributes can use single or double quotes

```
<p id="a123" class='shaded'>Text!</p>
```

# HTML SYNTAX (4/5)

- You can insert comments into your HTML document as well
  - Comments aren't rendered in the browser
    - They are there for developers to make notes of important things
      - e.g., what the element is for – news section
  - Comments can appear almost anywhere
    - e.g., before the root element, after the root element, in the head/body
    - Only place they can't is inside of attributes and tags

```
<!--
  This section is for the news feed from Twitter.
  -->
```

# HTML SYNTAX (5/5)

- What about inserting characters that are part of the HTML syntax (e.g., '<')?
  - Use entities
    - entity syntax is `&name_of_entity;`
      - Ampersand, name of entity, semi-colon
  - Common entities
    - < > " ' & ©
  - Can also use the Unicode values for the characters:
    - Ⓒ ™ Ⓡ —

# HTML STRUCTURE & CONTENT

- HTML documents
  - Are simply text files
    - Which can be created with a text editor (e.g., Brackets, Visual Studio Code, Atom.io)
  - Contain exactly one root element: html
  - The html element contains exactly two child elements:
    - head, body
  - Whitespace in HTML is collapsed
    - E.g., 2 spaces, 20 tabs, 4 new lines = 1 space

# HTML ELEMENTS

- The root element is the `html` element
  - And has the lang attribute (set to "en" for English)
    - There are a whole set of languages supported
- The `head` element is first child of `html`
  - Contains non-rendered elements
  - And contains several children of its own:
    - `title`, `link`, `meta`, `script`, `style`
      - We'll take a look at the most commonly used ones
- The `body` element is the second child element of `html`
  - and it contains all of the renderable elements that are displayed in a web browser tab/window

# HTML HEAD ELEMENTS (1/2)

- The `title` element
  - Used to display text in the Web browser's title bar – found in the head element
- The `link` element offers a way to link to other documents
  - Most commonly Cascading Style Sheets (CSS)
    - Something we'll look at later on in the course
- The `meta` element
  - Used to provide metadata about the page (e.g., character set used)
  - Used to be used for SEO, but is no longer acknowledged by search engines

# HTML HEAD ELEMENTS (2/2)

- The `style` element
  - Allows for the embedding of style information within the HTML document itself
    - This should be avoided because:
      - One style sheet can be linked to by multiple HTML documents
        - Thereby cutting down on the amount of time it takes to download content for the page
      - It also allows the web browsers to download multiple documents at once
        - Thereby speeding up the process of constructing the page

# HTML BODY ELEMENTS (1/2)

- The body element contains all of the visible elements that are rendered by the web browser
- Common body elements:
  - Heading elements (h1 – h6)
  - p element – for paragraphs
  - br element – for creating a newline or space between sections
  - ul element – creates unordered (i.e., bulleted lists)
  - ol element – creates ordered lists (e.g., 1, 2, 3 …, i, ii, iii, iv …)
  - dl element – creates definition lists (with term and definition pairs)

# HTML BODY ELEMENTS (2/2)

- Common body elements (cont'd):
  - Paragraph fragment elements:
    - span element – a section of text (usually in a paragraph)
    - i element – for italic text (usually in a paragraph)
    - b element – for bold text (usually in a paragraph)
    - a element – anchor, hyperlink linking to other places within the existing document
      - or to another document entirely
    - em element – for creating emphasis
    - small element – creates text that is smaller
    - abbr element – for abbreviations; creates a 'tooltip' for the abbreviation

# LISTS

- Lists come in three different flavors:
  - Ordered list
    - `ol` = ordered list, `li` = list item
  - Unordered list
    - `ul` – unordered list
  - Definition list
    - `dl` = definition list, `dt` = definition term, `dd` = definition description

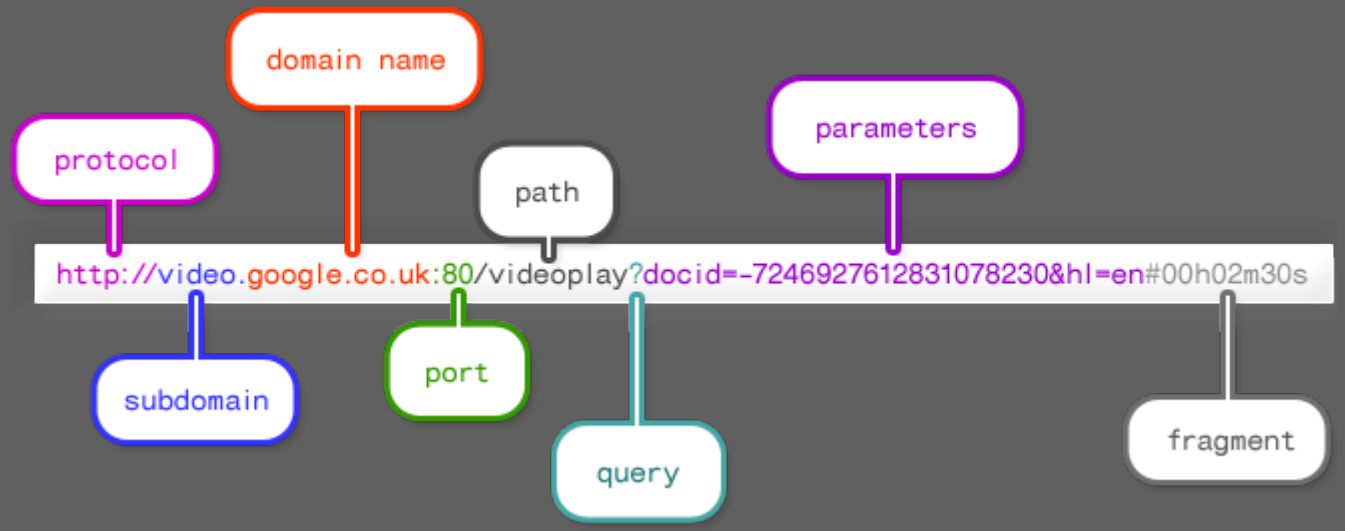
# HTML 5 & PATHS (1/3)

- Within the context of Web:
  - A path is a syntax that references a resource using a hierarchical syntax
  - A path can refer to a file
    - E.g., image, CSS file, JavaScript script, PDF document
  - A resource that is generated programmatically
    - E.g., a video, a page, an image, a PDF document
- Path syntax uses:
  - Front slash '/' for delimiters between sub sections of resources
  - '.' for current position within a hierarchy
  - '..' for referring to a super section
  - '#' for referring to a named anchor within an HTML document

# HTML 5 & PATHS (2/3)

- Paths can be:
  - Relative to the resource referring, so use '.' or '..' at the beginning
  - Absolute: referring to the full resource with it's full name
- Parts of a Uniform Resource Locator (URL):
  - Protocol, colon, two front-slashes, domain name, port, path, query, fragment



# HTML 5 & PATHS (3/3)

- Rules of thumb:
  - Use relative paths for referring to resources on your own site
  - Use absolute paths for referring to resources outside of your site
  - Paths are case-sensitive (i.e., exact spelling is required)
  - Create separate sub-directories for things like:
    - Images, CSS files, JavaScript scripts, PDF documents
  - Remember that '/' is different from './'
  - That '.' says current section within the hierarchy
  - That '..' says go up a section within the current hierarchy
  - Avoid creating resources that have spaces in the name (use dashes or underscores)
  - Remember to put quotes (either double or single) around your paths

# HTML TABLES (1/2)

- The table element:
  - Is for tabular data (rows and columns)
  - Historically got misused for creating layouts in the early days of web
    - due to the limited set of layout choices of HTML and CSS
- Tables have:
  - a caption (via the 'caption' element), optional
  - table rows (tr element) which contain either
    - td – table data
    - th – table header
  - partitioning with table head (thead), table footer (tfoot), and table body (tbody) – these are optional as well

# HTML TABLES (2/2)

- The thead, tfoot, and tbody are useful with CSS
  - in order to make the body scrollable but not the header or footer
- Making cells span columns
  - Column spans, use the `colspan` attribute
  - Row spans, use the `rowspan` attribute

# IMAGES

- In order to display images, use the img element
  - src attribute – link to the image to use
  - alt attribute – give quick description in case image cannot/won't be rendered
- Image elements can go inside of many different elements
  - paragraphs, list items, table cells, etc.
- Image file formats supported by all browsers:
  - GIF, PNG, JPEG

# FORMS AND INPUTS (1/2)

- The form element is one of the most important ways for communication to a web server
- Used for many different reasons:
  - Log users into a web app
  - Submit a transaction
  - Upload a photo
  - Edit a user profile
  - Play an online game
- Forms can have an 'action' attribute
  - Which contains the URL to the resource on the server that will respond

# FORMS AND INPUTS (2/2)

- Some of the child elements of forms
  - Input – for submitting data into the form
  - Text area – for large amounts of textual data
  - Select – for drop down lists or scrollable lists
  - Field set – for grouping inputs, text areas, and selects together
    - Can have multiple groupings for large forms (e.g., core contact info, avatar, history, etc.)
    - Field set can have a legend (i.e., a label)

# THE INPUT ELEMENT (1/2)

- Versatile element that renders differently based on its type attribute:
  - button, checkbox, color, date, datetime, datetime-local, email, file, hidden
  - image, month, number, password, radio, range,
  - reset search submit tel text time url week
- Other input element attributes:
  - autofocus – the one field in the form that is automatically selected for input
  - checked – for radio and checkbox; represents selection
  - disabled – cannot be interacted with
  - min/max – the minimum and maximum values respectively for numbers/dates
  - list – refers to a data list for the input (i.e., predefined options)

# THE INPUT ELEMENT (2/2)

- Other input element attributes (cont'd):
  - multiple – allows multiple files to be selected for type email and file
  - name – the name of the input (useful for JavaScript)
  - pattern – a regular expression defining the type of character data allowed
  - placeholder – visual hint for what the input expects
  - readonly – does not accept input but can be interacted with
  - required – user is required to supply data for this input
  - step – the step value for number intervals (e.g., 2)
  - value – the value for the input

# PLUGIN ELEMENTS – WHAT TO USE? (1/2)

- The `object` element
  - Nested browser content, legacy
- The `embed` element
  - New to HTML (only in HTML5) but has been supported by browsers for quite some time
- The `audio` element – for audio clips
- The `video` element – for video clips
- The `iframe` element – nested browser content

# PLUGIN ELEMENTS – WHAT TO USE? (2/2)

- The `iframe` element
  - For everything
    - YouTube videos, audio files, PDF documents, videos, SVG, everything
  - Works with web frameworks
    - E.g., Twitter's bootstrap
  - Allows elements inside (e.g., scripts)
  - Supported by all current browsers

# THE CANVAS ELEMENT

- For drawing … stuff
  - Creates a drawing context
  - Can be 2D
  - Can be 3D
    - Uses WebGL (GL = graphics library … from OpenGL)
    - Allows JavaScript to talk directly to the graphics processing hardware (i.e., graphics card)

# THE DIV AND SPAN ELEMENTS

- The div element allows the grouping of child elements within it
  - This grouping is logical and has nothing to do with the structural semantics – although it can
- The span element groups text content within a paragraph
  - Which can later have specific style added to it
- However, we cannot appreciate what the div and span elements do without visiting CSS … Which is for next week!

# SEMANTIC ELEMENTS (1/2)

- section – defines a section of the page
- header – the header of the page
- footer – footer of the page
- nav – navigation portion of page
- article – primary content of page
- aside – extra content (e.g., sidebar)
- figure – annotated images for content

# SEMANTIC ELEMENTS (2/2)

- mark – section of text that is 'marked'
- time – content that is time/date related
- meter – content is a fraction (e.g., disk usage)
- progress – indicate progress of a task

# HTML 5

- HTML rebooted:
  - New form elements (e.g., date, color, email)
  - Persistent local storage (for multiple tabs)
  - Session storage (for sessions)
  - Websocket (sending/receiving streaming data)
  - Canvas – drawing stuff fast (2D/3D)
  - Audio/video – so plugins aren't needed
  - Geolocation – share physical location
  - Drag-n-drop – create intuitive interfaces
  - Custom data – add custom attributes to elements

# HTML STRUCTURE & CONTENT

- HTML documents
  - Are simply text files
    - Which can be created with a text editor (e.g., Brackets, Visual Studio Code, Atom.io)
  - Contain exactly one root element: html
  - The html element contains exactly two child elements:
    - head, body
  - Whitespace in HTML is collapsed
  - E.g., 2 spaces, 20 tabs, 4 new lines = 1 space

# CHECKLIST FOR EDITOR/BROWSER

- Many editors have come and gone or are outdated. Some that have gone:
  - ~~Brackets~~, Notepad++, Dreamweaver, Atom, TextWrangler (Mac OS), Sublime – countless others!
- We'll use: Either Brackets or Microsoft Visual Studio Code
- Google Chrome
  - Mozilla Firefox okay
  - Don't use Microsoft Internet Explorer/Edge or Apple Safari
    - They will slow you down
- Utilize developer tools in the browser for debugging all your code
  - HTML, CSS, JavaScript

# CHECKLIST FOR HTML 5 WEB PAGE

- html is your root element
- The html element has both one head and one body element (and in that order)
- The head element has title as its first child element
- The head element has a meta element with the charset attribute
- The body contains all of the renderable elements (e.g., p, ul, etc.)
- All elements follow the LIFO rule:
  - Last one in, first one out for the begin tag/end tag rules

# CHECKLIST FOR EDITING HTML DOCUMENTS

- Indent your code
  - spaces are preferred – don't use tabs
    - Your tab size may be different than others
  - two spaces per tab is a common format for HTML, CSS, and JavaScript
    - JavaScript also uses four spaces per tab
- Save your work often
- Validate your work often (use: https://en.rakko.tools/tools/58/ the validator tools in your text editor)
  - In Microsoft Visual Studio Code go to the gear/settings -> Trust workspace folder(s)
    - File->preferences->Extensions->HTMLHint by Mike Kaufman

# RESOURCES

HTML Tutorial

- https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Introduction

HTML Element Reference

- https://developer.mozilla.org/en-US/docs/Web/HTML/Element