1. HTML = HyperText Markup Language
   1. Allows programs to **STRUCTURE** content on a website using tags.
      1. **Tags** =
   2. HTML5 is the current version
   3. Basic Anatomy of a webpage:
      1. <!DOCTYPE>
      2. <HTML>
         1. <HEAD><HEAD>
         2. <BODY><BODY>
      3. <HTML>
   4. **Elements** 🡺 root element 🡺 Node = content between tags + tags
      1. Elements are fundamental to HTML functionality
2. CSS = Cascading Style Sheets
   1. Complementary to HTML and provides the **AESTHETIC** (style) to HTML webpages
      1. Properties
         1. Styling precedence based on “ways to style”: Inline > Internal > External > Browser default
         2. 3 ways to style (in order from best practice to ‘worst’)
         3. External = includes a reference to a .css file inside the <head>
            1. Separation of concerns by separating styling from structure
            2. Reusability
            3. Centralization of styling
            4. Improves developer readability by reducing cluttering
         4. Internal = styling is defined in a style element inside the <head>
         5. Inline = styling is applied to each element via a **style attritbute**
      2. Str
         1. Adds style and flavor to the HTML pages
      3. Lim
   2. Syntax
      1. Selector
      2. Declaration
      3. Properties
      4. Value
      5. Statements
   3. Selectors
      1. Properties
      2. Str
      3. Lim
   4. Inheritance
      1. Properties
      2. Str
      3. Lim
   5. Colors
      1. Properties
      2. Str
      3. Lim
   6. Units of Measurement
      1. Properties
      2. Str
      3. Lim
   7. Specificity
      1. Properties
      2. Str
      3. Lim
   8. Pseudo
      1. PSelectors
      2. PClasses = keyword at the end of a CSS selector to specify a style *iff* the element is in a certain state
         1. Properties
            1. 30+ pseudo classes
         2. Str
            1. Allows for more precise styling dependent on business needs
         3. Lim
      3. PElements = styles specific parts of an element (for example the first line of a paragraph)
         1. Properties
         2. Str
         3. Lim
   9. Combinators = describes the relationship between selectors in a CSS statement
      1. Properties
         1. 4 types of combinators
            1. Descendant
            2. Child
            3. Sibling
            4. General Sibling
      2. Str
      3. Lim
   10. Box Model
       1. Properties
       2. Str
       3. Lim
3. MVC = An architectural pattern that creates applications with a 3 layer design: Model – View - Controller
   1. Properties
      1. Models = the virtual entities that exist in the application as stand-ins for the Db
         1. Model classes user validation logic to enforce **business rules** about the data – typically retrieving and storing model state
      2. View = *components*of the UI that users interact with – depending on the model data it “illustrates”
      3. **Controller** = the centerpiece of the structure that controls the flow of data (the entry point of the application)
         1. Do not have logic – think of gateways, but no control over when they open or close
         2. Have different information that flows through:
            1. Browser requests (HTTP requests)
            2. Model state data
            3. View templates for certain model states
      4. Conceptually, V C are different layers but in the same project
         1. M is a conceptual layer that is a functional amalgamation of the business logic, models, and context (Db access)/repo
   2. Separations of concerns =
      1. Properties
         1. Applications should be built with loosely coupled layers that collaborate to achieve specialized concerns
         2. Utilizes **layers** – which cannot communicate beyond the scope of their neighbor
      2. Str
         1. Allows for loosely coupled code
         2. Not unique to .NET – means that robust understanding of MVC can allow you to understand program design architecture of MVC in other languages
      3. Lim
         1. A separate architectural design from other common patterns:
   3. Repository layer
   4. Repository pattern
   5. Decoupling
   6. Model = represents the state of the application and any business logic
      1. Properties
         1. Contains two types of logic:
            1. Implementation logic
            2. Business logic
      2. Str
      3. Lim
   7. View = responsible for presenting content to an interface (These are the html files that will actually display the views)
   8. Controller =
      1. Properties
         1. Should have minimal logic that interacts with Model layer and View layer
         2. Invokes action methods if the HTTP endpoint matches the incoming URL
            1. Default routing logic is: Broad 🡺specific
            2. /[Controller]/[ActionName]/[Parameters]
      2. Str
      3. Lim
   9. ASP.NET MVC = **HIGHLY TESTABLE**
      1. Properties
         1. Should have minimal logic that interacts with Model layer and View layer
         2. S-N: Recall that each aspect of MVC is a layer and already “compartmentalized out” for you
      2. Str
         1. **HIGHLY TESTABLE**
         2. Easy to update compared to other patterns (Example:)
         3. A patterns-based way to ensure separation of concerns
      3. Lim
   10. Routing data =
   11. Routing logic
       1. Attribute routing
       2. Conventional routing