Agenda

- Web Architecture
- Introduction to HTML
- Tags

Application

- It is a program that contains set of instructions for the CPU
- It can be developed in any type of langugage
- Their are generally 2 types of application

1. Native Application
    - Developed using C and CPP
    - It is bit faster
    - It is OS Dependent
2. Web Application
    - Developed using HTML,CSS and JavaScript
    - slower than the native application
    - It is OS Independent

Server

- It is a physical device with highest configuration that is used to process the data on large scale and give out the response
- It generally consists of a-

1. Web server
2. DB Server
3. Languages
4. Operating System (platform)

- Server can have a Software Stack which is a Collection of softwares
- examples

1. WISA
    - Windows
    - IIS
    - SQL Server
    - ASP.net
2. MEAN
    - MongoDB
    - Express
    - Angular
    - Node

Browser Architecture

1. User Interface

- This is the user interface for the browser.
- It includes the Address Bar, back button, Bookmarking options, Refresh button, etc.
- The browser's user interface is not specified in any formal specification, but comes from practices shaped over decades of experience (and browsers copying each other).
- As a result, all browsers have UIs that are extremely similar to each other.

2. The Browser Engine

- The browser engine marshals actions between the browser's user interface and the browser's rendering engine.
- When you type in a new website and press the enter key, the browser UI will tell the browser engine, which will then communicate with the rendering engine.

3. The Rendering Engine

- The rendering engine is responsible for displaying the requested content.
- The rendering engine will start by getting the contents of the requested document from the networking layer.
- It takes in the HTML code and parses it to create the DOM (Document Object Model) tree.
- Examples of rendering engine include
    1. Safari - WebKit Rendering Engine
    2. Chrome - Blink Rendering Engine (Blink is a fork of WebKit)
    3. FireFox - Gecko Rendering Engine

4. Networking Layer

- The Networking Layer is responsible for making network calls to fetch resources.
- It imposes the right connection limits, formats requests, deals with proxies, caching, and much more.

5. JavaScript Engine

- The JavaScript Engine is used to parse and execute JavaScript code on the DOM.
- The JavaScript code is provided by the web server, or it can be provided by the web browser
- Early browsers used JavaScript interpreters, but modern JavaScript engines use Just-In-Time compilation for improved performance.
- Examples of JavaScript Engine include
    1. Safari - JavaScriptCore
    2. Chrome - V8 JavaScript Engine
    3. FireFox - SpiderMonkey Engine

6. UI Backend

- This layer is responsible for drawing the basic widgets like select or input boxes and windows. Underneath it uses operating system UI methods.
- The rendering engine uses the UI backend layer during the layout and painting stages to display the web page on the browser.

7. Data Storage

- The browser needs to save data locally (cookies, cache, etc.) so the Data Storage component handles this part.

Web Architecture

1. Web Server

- A web server is a system (hardware and/or software) that delivers web content, such as websites, web pages, and other resources, to users over the internet or an intranet.
- It handles requests from client devices, typically browsers, and serves the requested resources using standard web protocols like HTTP (Hypertext Transfer Protocol) or HTTPS (HTTP Secure).
- There are many web server software options, each with unique features and use cases. Popular ones include:
    1. Apache HTTP Server: Open-source and widely used, known for flexibility and extensive module support.
    2. Nginx: High-performance and lightweight, often used for handling large amounts of traffic.
    3. Microsoft IIS (Internet Information Services): A web server for Windows environments, integrated with Microsoft technologies.

2. HTTP Request

- An HTTP request is sent by a client to a server to ask for a resource or perform an action.
- It consists mainly of two parts - Header and Body
- Headers Provide additional information about the request.
- Examples:
    - Host: Specifies the target host (e.g., Host: example.com).
    - User-Agent: Identifies the client making the request (e.g., browser or app).
    - Content-Type: Indicates the format of the request body (e.g., application/json).
- Body (Optional): Contains data sent to the server (used in methods like POST or PUT).

3. HTTP Response

- An HTTP response is sent by the server to the client in reply to a request, containing the requested resource, a status code, and other metadata.
- It consists mainly of two parts - Header and Body
- Headers Provide metadata about the response.
- Examples:
    - Content-Type: The format of the response data (e.g., text/html or application/json).
    - Content-Length: The size of the response body in bytes.
    - Set-Cookie: Used to send cookies to the client.
- Body (Optional): Contains the requested resource or additional information.
- Example:
    - HTML for a webpage.
    - JSON data for an API.

HTTP Request Methods

1. GET: Requests a resource without modifying it.
2. POST: Submits data to the server for processing.

3. PUT: Updates or replaces an existing resource.
4. DELETE: Deletes a resource.

HTTP Response Status Code

1. 1xx (Informational): Request received, continuing process.
    - 100 Continue: Initial part of a request received.
2. 2xx (Success): Request was successful.
    - 200 OK: Request succeeded.
    - 201 Created: Resource was successfully created.
3. 3xx (Redirection): Client needs to take further action.
    - 301 Moved Permanently: Resource has a new URL.
    - 302 Found: Resource temporarily moved.
4. 4xx (Client Errors): Issues with the client's request.
    - 400 Bad Request: The request is malformed.
    - 401 Unauthorized: Authentication required.
    - 404 Not Found: Resource not found.
5. 5xx (Server Errors): Issues on the server side.
    - 500 Internal Server Error: Generic server error.
    - 503 Service Unavailable: Server is overloaded or down.

HTML

- HTML (HyperText Markup Language) is the standard language used to create and structure content on the web.
- It provides the foundation for web pages by defining the structure and elements of the content, such as headings, paragraphs, links, images, tables, and more.
- The latest standard is HTML5
- To add html 5 code, start the document with

```
<!DOCTYPE html>
```

- DOCTYPE: document type (tag used to start the html document)
- It is Case in-sensitive
- To add comment

```
<!-- comment -->
```

Key Features of HTML

1. Markup Language: HTML uses "tags" to mark up different parts of content, indicating their roles (e.g., heading, paragraph, list).
2. HyperText: Enables linking to other web pages or resources using hyperlinks (`<a>` tag).
3. Platform-Independent: HTML can be rendered on any device with a web browser.

4. Extensible: Can be combined with other technologies like CSS (for styling) and JavaScript (for interactivity).

HTML History

1. HTML 1.0 (1993)

- Tim Berners-Lee, the inventor of the World Wide Web.
- Purpose was to share research documents and enable hyperlinking between them.
- Key Features:
    - Basic text formatting (headings, paragraphs, lists).
    - Hyperlinks (`<a>` tag).

2. HTML 2.0 (1995)

- Standardize HTML for broader use.
- Key Features:
    - Support for tables and forms.
    - Basic image embedding (tag).
    - Standardized syntax rules.

3. HTML 3.2 (1997)

- Managed by the World Wide Web Consortium (W3C).
- Introduced features for richer content.
- Key Features:
    - Support for scripting languages like JavaScript.
    - Improved table support for better layouts.
    - Introduction of new elements like for styling.

4. HTML 4.01 (1999)

- W3C continued to refine HTML.
- Enhanced web functionality and accessibility.
- Key Features:
    - Separation of content and style via CSS.
    - Introduction of attributes like id and class for better styling and scripting.
    - Improved support for forms and multimedia.

5. XHTML 1.0 (2000)

- W3C attempted to merge HTML with XML.
- Introduced stricter syntax and better compatibility.
- Key Features:
    - All tags must be properly closed (e.g.,
).
    - Case sensitivity for tags and attributes.
- Challenges:
    - Too strict for practical use; adoption was limited.
    - Developers found it difficult to migrate from HTML 4.

6. HTML5 (2014, Official Recommendation)

- Modernize HTML for web applications and multimedia.
- Key Features:
  - Semantic Elements: `<header>, <footer>, <article>, <nav>` for better content structure.
  - Multimedia Support: Native audio (`<audio>`) and video (`<video>`) elements.
  - Canvas and SVG: For drawing graphics and animations directly in the browser.
  - APIs: Built-in APIs for features like geolocation, drag-and-drop, and offline storage.
  - Backward Compatibility: Maintains support for older HTML versions.
- It provided Simplified coding practices.

Tags

- Word enclosed by < and > signs
- It is also called as an element
- All tags in HTML are pre-defined by W3C
- E.g. `<h1>, <p>, <table>`

Types of tags

1. Opening
   - Used to open a data/information
   - E.g. `<h1>, <p>, <html>`
2. Closing
   - Used to close the data/information
   - E.g. `</h1>, </p>, </html>`
3. Empty
   - Tag having no data/content
   - Two ways of representing it
     - `<tag></tag>`
     - `<tag />`
     - E.g. `<br/>, <hr/>`
4. Root
   - Tag which starts and ends the document
   - Is also called as Document Type or Document Tag or Document Element
   - E.g. `<html>` is root tag for html document