

Agenda

- Basic Introduction
- Web Architecture

Application

- It is a program that contains set of instructions for the CPU
- It can be developed in any type of language
- There 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

- A web browser is a software application that allows you to access, locate, and display information, such as web pages, images, and videos, on the World Wide Web
- Components of Browser Architecture are

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

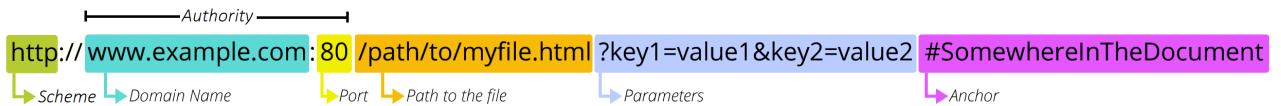
- An HTTP request is sent by a client to a server to ask for a resource or perform an action.
- It consists mainly of two parts - Header and Body
- Headers Provide additional information about the request.
- Examples:
 - Host: Specifies the target host (e.g., Host: example.com).
 - User-Agent: Identifies 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.

URL

- A URL (Uniform Resource Locator) is the address of a unique resource on the internet. It is one of the key mechanisms used by browsers to retrieve published resources, such as HTML pages, CSS documents, images, and so on.
- A URL is composed of different parts, some mandatory and others optional.



1. Scheme

- The first part of the URL is the scheme, which indicates the protocol that the browser must use to request the resource (a protocol is a set method for exchanging or transferring data around a computer network).
- Usually for websites the protocol is HTTPS or HTTP (its unsecured version)

2. Authority

- Next follows the authority, which is separated from the scheme by the character pattern 😕/.
- If present the authority includes both the domain (e.g., www.example.com) and the port (80), separated by a colon:
- The domain indicates which Web server is being requested.
- Usually this is a domain name, but an IP address may also be used (but this is rare as it is much less convenient).
- The port indicates the technical "gate" used to access the resources on the web server.
- It is usually omitted if the web server uses the standard ports of the HTTP protocol (80 for HTTP and 443 for HTTPS) to grant access to its resources. Otherwise it is mandatory.

3. Path to resource

- /path/to/myfile.html is the path to the resource on the Web server.
- In the early days of the Web, a path like this represented a physical file location on the Web server.
- Nowadays, it is mostly an abstraction handled by Web servers without any physical reality.

4. Parameters

- ?key1=value1&key2=value2 are extra parameters provided to the Web server.
- Those parameters are a list of key/value pairs separated with the & symbol. - The Web server can use those parameters to do extra stuff before returning the resource.

5. Anchor

- #SomewhereInTheDocument is an anchor to another part of the resource itself.
- An anchor represents a sort of "bookmark" inside the resource, giving the browser the directions to show the content located at that "bookmarked" spot.- On an HTML document, for example, the browser will scroll to the point where the anchor is defined; on a video or audio document

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.