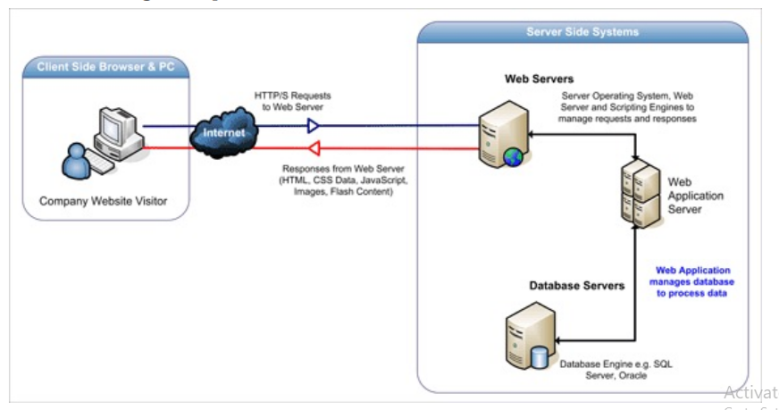
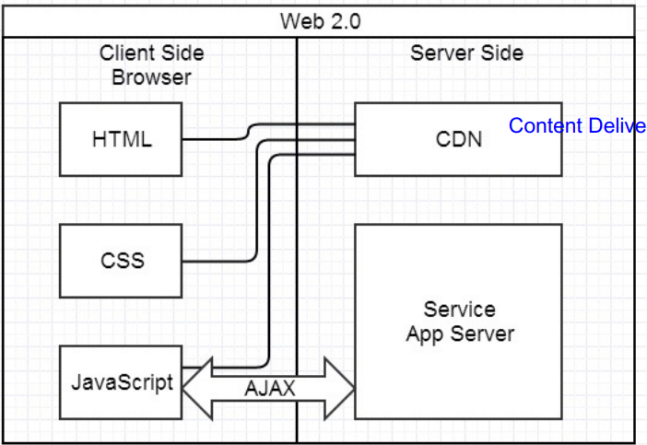
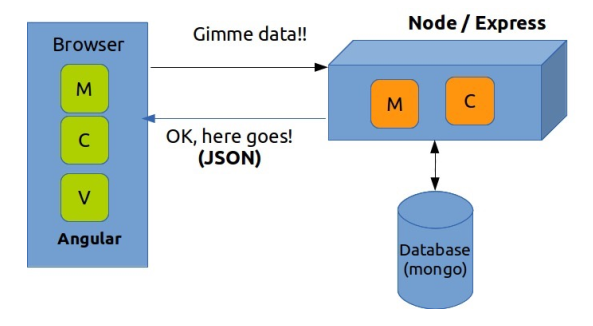
Web Application - web server







**Apache** - web server software to **host/configure** dynamic web pages, static files, and other content over the internet.

Web browser is an example of a user agent (UA)

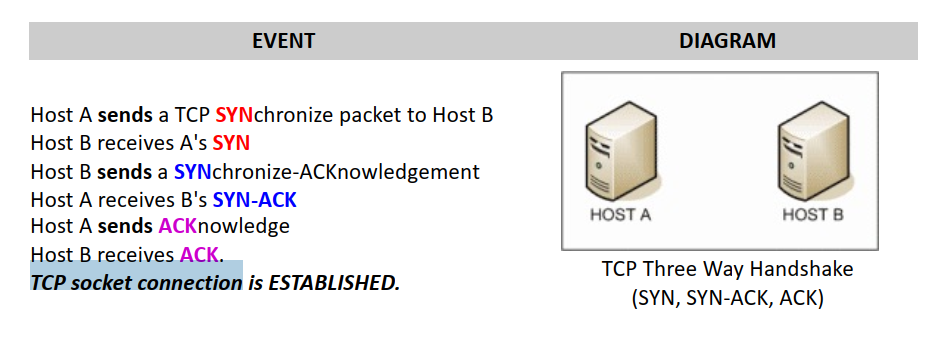
WebSockets are a communication protocol that enables real-time, two-way communication between a client and a server over a single, long-lived connection.

**HTTP** stands for Hypertext Transfer Protocol. It is an application layer protocol that is used for transmitting data over the World Wide Web (WWW). thee protocol is based on a client-server model, where the client sends a request to the server, and the server responds with the requested data. The request typically consists of a request method (such as GET, POST, or PUT), a Uniform Resource Identifier (URI) identifying the resource to be accessed, and optional headers that provide additional information about the request**.HTTP uses TCP/IP as its transport protocol**, which means that it establishes a reliable, connection-oriented communication channel between the client and the server.

**HTTP vs TCP**

TCP (Transmission Control Protocol) is a transport layer protocol that provides reliable, ordered, and error-checked delivery of data between applications running on different devices over a network.It is responsible for breaking down data into packets, ensuring that they are delivered in the correct order, and retransmitting any lost or corrupted packets to ensure that all data is delivered successfully.

HTTP (Hypertext Transfer Protocol), on the other hand, is an application layer protocol that governs the way web clients (such as web browsers) and servers communicate with each other. **It uses TCP as its underlying transport protocol to establish** a connection and exchange data between a client and a server.



Yes, **HTTP (Hypertext Transfer Protocol) is a stateless protocol**. This means that the server does not maintain any information about previous requests made by the client. Each request made by the client is considered a new and independent request, and the server treats it as such.

However, it also means that HTTP is not well-suited for certain types of applications that require maintaining state information between requests, such as real-time collaborative applications or online games. To handle these scenarios, additional technologies such as **WebSockets** or server-side state management may be necessary.

So, typically, the TCP Three-Way Handshake **is only performed onc**e at the beginning of the connection. After that, subsequent HTTP requests and responses can be sent and received using the established connection without the need to perform the handshake again.

**A cookie (also known as an HTTP cookie)** is a small text file that is sent by a web server to a web browser, and is stored by the browser. Cookies are commonly used to store information about a user's preferences, login status, or browsing history on a particular website.

When a user visits a website, the web server may set one or more cookies on the user's browser. The browser stores the cookies and **sends them back to the server with subsequent request**s. This allows the server to recognize the user and provide personalized content, s**uch as remembering the user's language preference or displaying targeted advertisements.**

The cookie is then stored in your browser's **cookie jar**, which is a collection of cookies associated with a specific domain. When you make subsequent requests to the same website, your **browser sends the cookies associated with that domain back to the web server in the Cookie header of the request**. This allows the web server to retrieve information about your previous interactions with the website, such as your login status, preferences, or items in your shopping cart.

Cookies can be either "session" cookies or "persistent" cookies. Session cookies are temporary and are deleted when the user closes their browser. Persistent cookies, on the other hand, remain on the user's device even after the browser is closed, and can be used to remember the user's preferences across multiple sessions.

**localStorage vs sessionStorage js**

localStorage and sessionStorage are both client-side storage options available in modern web browsers that allow developers to store data on the client's computer.

The main difference between the two is the lifetime of the data they store.

localStorage: The data stored using localStorage persists even after the browser is closed and reopened.

sessionStorage: The data stored using sessionStorage is only available for the duration of the current browser session. Once the browser is closed, the data is deleted.

When data is stored using localStorage, it is saved a**s key-value pairs** in a local database on the user's computer.

Google Chrome: C:\Users\[user]\AppData\Local\Google\Chrome\User Data\Default\Local Storage

**Application Cache,** also known as AppCache, is a web browser technology that allows web developers to make web applications available offline by caching the files needed to run the application in the user's browser.

When a user visits a web page that uses AppCache, the browser downloads and caches the files required to run the application, including HTML files, JavaScript files, CSS files, images, and other resources. The next time the user visits the same web page, the browser can load the files from the cache instead of downloading them again, which makes the application load faster and work offline.

AppCache uses a manifest file, which is a text file that lists all the files that need to be cached. The manifest file can also specify whether files should be cached only when the user is online or always, and whether the cached files should be updated when the user goes online again.

AppCache is being replaced by newer technologies such as Service Workers, which offer more flexibility and control over caching and offline functionality.

**Service Workers** are a web browser technology that allows web developers to write scripts that run in the background of a web page and intercept network requests made by the page. Service Workers can be used to implement advanced caching and offline functionality, push notifications, and background synchronization.

**Web Workers** are a web browser technology that allows web developers to write scripts that run in the background of a web page without blocking the main thread. This can be used to perform complex computations or long-running tasks without affecting the user interface of the web page.

**Server-sent events (SSE)** is a technology that enables a web page to receive automatic updates from a server via a persistent HTTP connection. It is a unidirectional communication channel where the server pushes data to the client without requiring the client to request the data.SSE is commonly used for real-time updates, such as social media feeds, stock prices, or sports scores.

**SE is a lightweight alternative to other real-time web technologies, such as WebSockets, which require a bidirectional connection.**

**DHTML (Dynamic HTML) vs AJAX (Asynchronous JavaScript and XML)**

While DHTML is focused on creating dynamic effects and interactivity within a single web page, AJAX is focused on improving the performance and user experience of web applications by reducing page reloads and improving the responsiveness of the application.

**JavaScript**

**let vs var**

var is scoped to the function or global scope and let is scoped to the block in which it is declared. It's generally recommended to use let

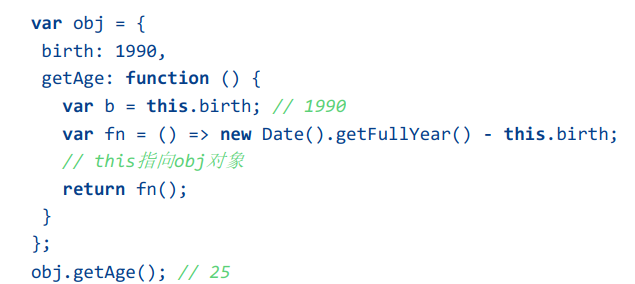
function addTwoNumbers(x, y) {

return x + y;

}

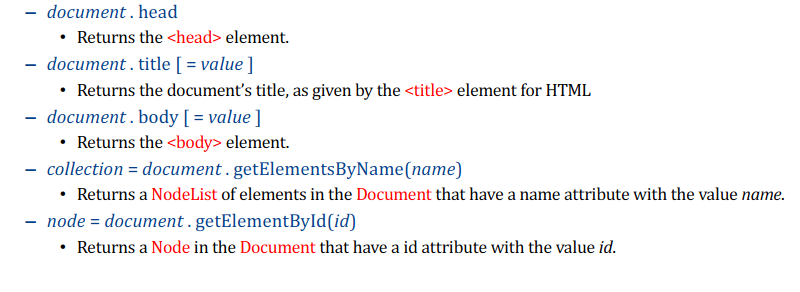
**lambda expression**

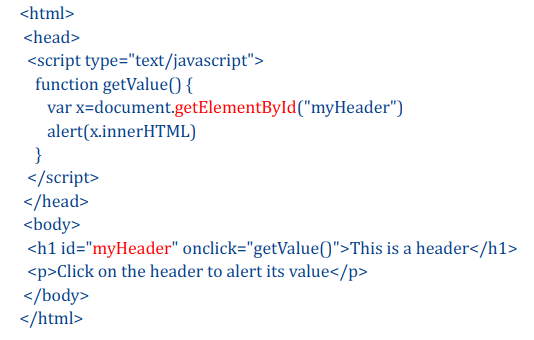
const x = (x, y) => x \* y; // ES6 (ES13 released in 2022)

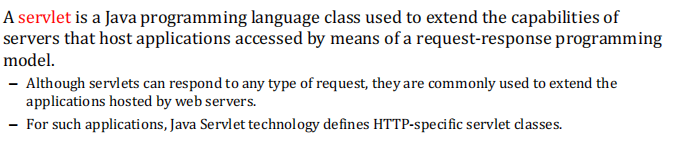


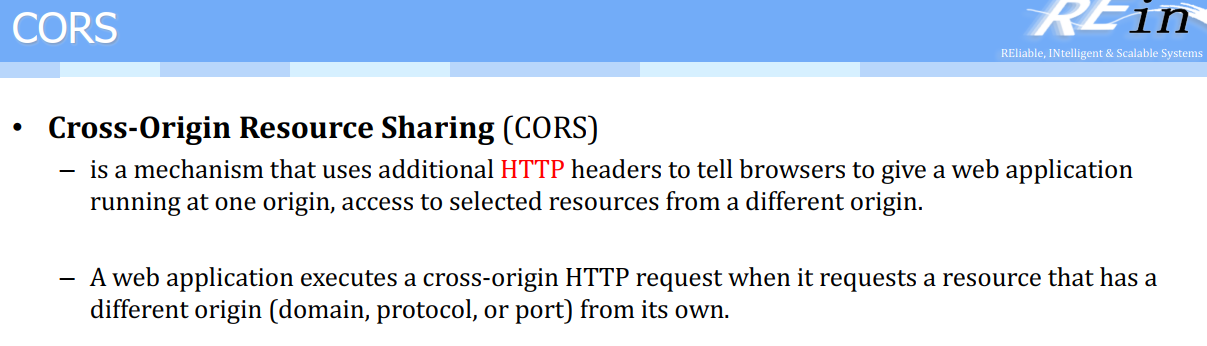
The **Document Object** is a core object in the web browser's JavaScript API that represents the HTML document displayed in the browser window. It provides an interface for JavaScript to interact with the document, including accessing and modifying its content, structure, and styling.

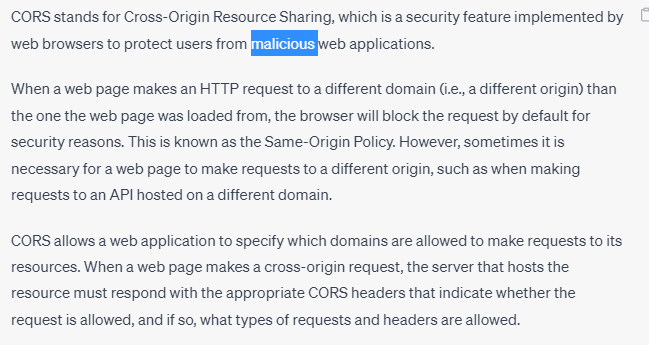
Every XML and HTML document in an HTML user agent is represented by a **Document object**.



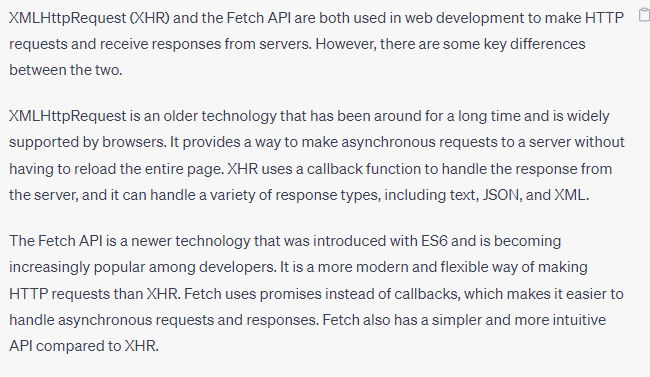


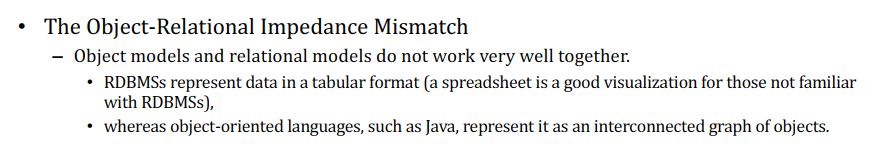


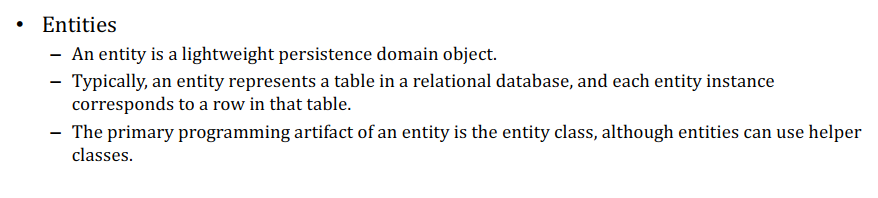


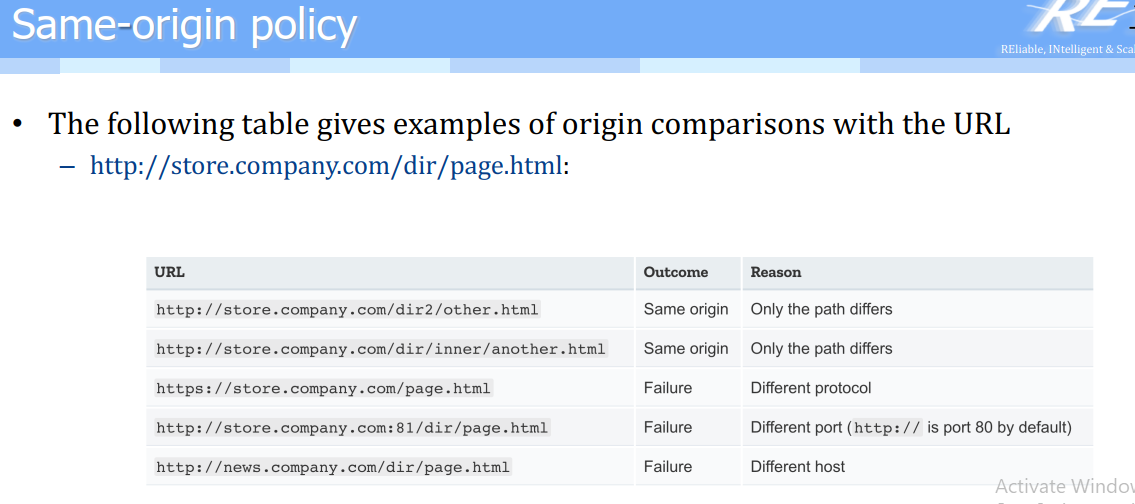


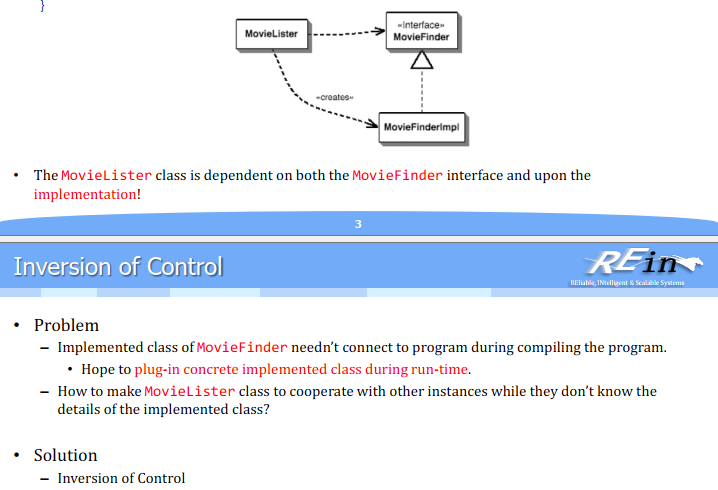
XMLHttpRequest (Ajax) vs the Fetch API











<https://www.tutorialsteacher.com/ioc/inversion-of-control>

