Web Development: History and Fundamentals

1. A Brief History of Web Development

The World Wide Web has evolved rapidly from simple text documents to complex, interactive applications.

The Origins (1989–1991)

- **1989:** Sir Tim Berners-Lee proposes the World Wide Web project at CERN to manage information sharing between scientists.
- **1990:** Berners-Lee writes the first web browser (**WorldWideWeb**) and the first web server. He defines the three fundamental technologies that remain the foundation of the web today:
  - **HTML:** Formatting language.
  - **URI (URL):** Unique address for resources.
  - **HTTP:** Protocol for retrieval.
- **1991:** The first website goes live.

The Browser Wars (1993–2001)

- **Mosaic (1993):** The first popular browser to display images inline with text, making the web accessible to non-technical users.
- **Netscape vs. Internet Explorer:** Netscape Navigator dominated early, but Microsoft aggressively integrated IE with Windows. This competition led to the rapid addition of features like JavaScript and CSS, often with compatibility issues.

The Dot-Com Bubble & Web 2.0 (2000–2010)

- **Web 2.0:** A shift from static "read-only" pages to dynamic "read-write" platforms. Users began generating content via blogs, wikis, and social media (e.g., Facebook, YouTube).
- **AJAX:** Allowed web pages to update asynchronously by exchanging data with a web server behind the scenes, leading to smoother applications like Google Maps.

The Modern Era (2010–Present)

- **Mobile-First:** With the rise of smartphones, Responsive Web Design (RWD) became standard to ensure sites worked on all screen sizes.
- **Single Page Applications (SPAs):** Frameworks like React, Vue, and Angular allow websites to feel like native apps, loading content dynamically without refreshing the page.

---

2. Introduction to HTML (HyperText Markup Language)

HTML is the standard markup language for documents designed to be displayed in a web browser. It provides the **structure** of a webpage.

Core Concepts

- **Tags:** HTML uses "tags" enclosed in angle brackets (e.g., <h1>) to define elements. Most tags come in pairs: an opening tag and a closing tag (e.g., </h1>).
- **Attributes:** Provide additional information about an element, usually placed in the opening tag (e.g., class, id, src).
- **DOM (Document Object Model):** The browser parses HTML into a tree structure called the DOM, which represents the page hierarchy.

Basic Document Structure

Every HTML document follows a standard boilerplate structure:

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Page Title</title>
</head>
<body>
    <h1>Welcome to My Website</h1>
    <p>This is a paragraph of text explaining the content.</p>
    <a href="https://example.com">This is a link</a>
</body>
</html>
```

Common Elements

Element	Description
<h1> to <h6>	Headings (h1 is the most important).
<p>	A paragraph of text.
<a>	Anchor tag used for hyperlinks. Requires an href attribute.
<img>	Embeds an image. Self-closing tag (no </img>).
	A generic container used to group elements for styling.
/	Unordered (bulleted) or Ordered (numbered) lists.
<!-- –>	Comment

---

3. Introduction to HTTP (HyperText Transfer Protocol)

HTTP is the protocol used for transmitting hypermedia documents, such as HTML. It follows a classic **Client-Server model**.

How It Works (The Request-Response Cycle)

1. **The Client (Browser):** Sends an **HTTP Request** to the server (e.g., "Get me the homepage for https://www.google.com/search?q=google.com").
2. **The Server:** Processes the request and sends back an **HTTP Response** containing the requested data (HTML, JSON, Images) and a status code.

HTTP Methods (Verbs)

Requests use specific "methods" to tell the server what action to perform:

- **GET:** Retrieve data from a server (e.g., loading a webpage).
- **POST:** Submit data to be processed to a server (e.g., submitting a login form).
- **PUT:** Update existing data on the server.
- **DELETE:** Remove data from the server.

HTTP Status Codes

The server response includes a 3-digit code indicating the result:

- **200 OK:** The request succeeded.
- **301 Moved Permanently:** The resource has moved to a new URL (redirect).
- **404 Not Found:** The server cannot find the requested resource.
- **500 Internal Server Error:** The server encountered a situation it didn't know how to handle.

HTTP vs. HTTPS

- **HTTP:** Data is sent in plaintext. If intercepted, it can be read by attackers.
- **HTTPS (Secure):** Uses TLS/SSL encryption. Data is encrypted between the client and server, ensuring privacy and security.