HTML

Getting Started with HTML

Definition:
HTML (HyperText Markup Language) is a standard markup language used to create and structure web pages.

Why / Purpose

- To define the structure and content of a web page
- Acts as the foundation of web development (CSS = style, JS = behavior)

Key Points

- HTML uses tags to define elements
- HTML files have .html or .htm extension
- HTML is not a programming language, it is a markup language
- Browsers interpret HTML and display content

Simple Example

```
<h1>Hello World</h1>
<p>This is my first web page</p>
```

Real-Life Example

HTML is like the skeleton of a human body – it gives structure, not style or actions.

Interview Notes

- HTML = Structure
- CSS = Design
- JavaScript = Functionality

HTML Document Structure

Definition:
HTML document structure defines the basic layout and required elements of an HTML page.

Why / Purpose

- Helps browsers understand and render the web page correctly
- Maintains standard and readable code

Key Points

- Every HTML page starts with <!DOCTYPE html>
- <html> is the root element
- <head> contains metadata
- <body> contains visible content

Basic HTML Structure

```
<!DOCTYPE html>
<html>
  <head>
```

```
    <title>My Page</title>
  </head>
  <body>
    <h1>Welcome</h1>
    <p>This is a webpage</p>
  </body>
</html>
```

Explanation of Tags

Tag	Purpose
<!DOCTYPE html>	Defines HTML5 version
<html>	Root element
<head>	Metadata (title, links, styles)
<title>	Page title shown on browser tab
<body>	Visible page content

Interview Notes

- <!DOCTYPE html> is mandatory
- Content inside <head> is not visible
- Only <body> content appears on the webpage

HTML5 New Features

Definition:
HTML5 is the latest major version of HTML that introduces new elements, attributes, APIs, and multimedia support without external plugins.

Why / Purpose

- To support modern web applications
- To reduce dependency on Flash, plugins, and external libraries
- To improve performance, accessibility, and user experience

Key Features of HTML5

- New semantic elements (<header>, <footer>, <section>)
- Multimedia support (<audio>, <video>)
- Graphics support (<canvas>, <svg>)
- Form enhancements (new input types & attributes)
- APIs (Geolocation, Local Storage, Web Workers)

Simple Example

```
<video controls>
  <source src="movie.mp4" type="video/mp4">
</video>
```

Real-Life Analogy
HTML5 is like a smartphone upgrade — same phone, but more features, faster, and smarter.

Interview Notes

- HTML5 does not require plugins

- <!DOCTYPE html> is simpler in HTML5
- Backward compatible with HTML

HTML5 Semantic Elements

Definition:
Semantic elements clearly describe their meaning and purpose to both browsers and developers.

Why / Purpose
- Improves code readability
- Helps SEO (Search Engine Optimization)
- Enhances accessibility

Common Semantic Elements

Element	Purpose
<header>	Top section / heading
<nav>	Navigation links
<section>	Related content
<article>	Independent content
<aside>	Sidebar content
<footer>	Bottom section

Example
```
<article>
  <h2>News</h2>
  <p>HTML5 released new features</p>
</article>
```

Interview Notes
- Semantic elements replace excessive usage
- Improves maintainability and SEO

HTML Elements

Definition:
An HTML element consists of a start tag, content, and an end tag.

Why / Purpose
- To structure and display content on a web page

Types of Elements
- Block elements → , <p>, <h1>
- Inline elements → , <a>, <img>
- Empty elements →
, <hr>, <img>

Example
```
<p>This is a paragraph</p>
```

Interview Notes

- Empty elements have no closing tag
- Block elements start on a new line

HTML Attributes

Definition:
Attributes provide additional information about HTML elements.

Why / Purpose
- To control behavior, appearance, and data of elements

Key Points
- Written inside opening tag
- Always in name="value" format
- Some attributes are global

Example
<input type="text" placeholder="Enter name" required>

Common HTML5 Attributes

Attribute	Purpose
required	Mandatory input
placeholder	Hint text
autofocus	Focus on load
readonly	Read-only field
disabled	Disable input

Interview Notes
- Attributes improve form validation
- HTML5 validation works without JavaScript

Important Interview Summary
- HTML5 = semantic + multimedia + APIs
- Elements define structure
- Attributes define behavior
- Semantic tags improve SEO & accessibility

HTML Headings

Definition:
HTML headings are used to define titles and subtitles on a web page.

Why / Purpose
- To give proper structure to content
- Helps SEO and readability
- Shows importance of content hierarchy

Key Points
- There are 6 heading levels: <h1> to <h6>

- <h1> is the most important
- Heading size decreases from <h1> to <h6>
- **Use only one <h1> per page (best practice)**

Example

```
<h1>Main Heading</h1>
<h2>Sub Heading</h2>
<h3>Section Heading</h3>
```

Real-Life Example

Headings are like chapter titles in a book.

Interview Notes

- Headings are block-level elements
- **Don’t use headings just for styling (use CSS)**

HTML Paragraphs

Definition:

The <p> tag is used to define a paragraph of text.

Why / Purpose

- To group related sentences
- Improves readability and content structure

Key Points

- <p> is a block element
- Automatically adds space before and after
- Does not preserve extra spaces or line breaks

Example

```
<p>HTML is used to create web pages.</p>
<p>This is another paragraph.</p>
```

Real-Life Example

Paragraphs are like paragraphs in an essay – one idea per block.

Interview Notes

-
 is used for line break inside a paragraph
- Avoid using multiple
 tags for spacing

HTML Links (Anchor Tag)

Definition:

HTML links are created using the anchor (<a>) tag to navigate between web pages.

Why / Purpose

- To connect multiple web pages
- Enables navigation on the internet

Key Points

- Uses href attribute to specify URL
- Can link to pages, files, emails, sections
- Can open link in same or new tab

Example

```
<a href="https://www.google.com">Go to Google</a>
```

Open Link in New Tab

```
<a href="page.html" target="_blank">Open Page</a>
```

Interview Notes

- <a> is an inline element
- target="_blank" opens a new tab
- href="#" is used as a placeholder link

HTML Images

Definition:

The <img> tag is used to display images on a web page.

Why / Purpose

- To make web pages visually attractive
- To represent information graphically

Key Points

- <img> is an empty (self-closing) tag
- Uses src attribute for image path
- Uses alt attribute for accessibility
- Inline element by default

Example

```
<img src="logo.png" alt="Company Logo" width="200">
```

Real-Life Example

Images are like pictures in a newspaper — they convey information quickly.

Interview Notes

- alt attribute is mandatory (SEO + accessibility)
- Image without alt is bad practice

HTML Lists

Definition:

HTML lists are used to display items in a structured manner.

Why / Purpose

- To organize content clearly

- Used in menus, navigation bars, steps

Types of Lists

🔹 **Ordered List (<ol>)**

Displays items with numbers.

```
<ol>
  <li>HTML</li>
  <li>CSS</li>
  <li>JavaScript</li>
</ol>
```

🔹 **Unordered List (<ul>)**

Displays items with bullets.

```
<ul>
  <li>Apple</li>
  <li>Banana</li>
</ul>
```

🔹 **Description List (<dl>)**

Used for terms and descriptions.

```
<dl>
  <dt>HTML</dt>
  <dd>Markup language</dd>
</dl>
```

Key Points

- is used inside <ol> and
- Lists are block-level elements
- Can be nested (list inside list)

Real-Life Example

Lists are like shopping lists or to-do lists.

Interview Notes

- Navigation menus are usually
- <dl> is less commonly used

HTML Tables

Definition:

HTML tables are used to display data in rows and columns.

Why / Purpose

- To show structured data
- Used in reports, marks, schedules

Table Tags

Tag	Purpose
<table>	Table container
<tr>	Table row
<th>	Table header
<td>	Table data

Example

```
<table border="1">
  <tr>
    <th>Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>John</td>
    <td>25</td>
  </tr>
</table>
```

Key Points

- <th> text is bold & centered
- Tables are block elements
- Use CSS for styling, not border attribute

Real-Life Example

Tables are like Excel sheets.

Interview Notes

- Tables are not for layout (use CSS)
- Use <thead>, <tbody>, <tfoot> for large tables

HTML Tag

Definition:

 is a block-level container element used to group HTML elements.

Why / Purpose

- To structure the layout of a web page
- Used for styling and positioning with CSS
- Helps in organizing content

Key Points

- Block-level element
- Starts on a new line
- Takes full width by default
- No visual effect without CSS

Example

```
<div class="container">
  <h2>Login</h2>
  <p>Enter your details</p>
</div>
```

Interview Notes

- Used heavily for page layout
- Often replaced by semantic tags in HTML5

HTML Tag

Definition:
 is an inline container element used to style or manipulate part of text.

Why / Purpose
- To apply CSS or JavaScript to small portions of content
- Does not break the line

Key Points
- Inline element
- Does not start on a new line
- Takes only required width
- No visual change without CSS

Example
```
<p>Welcome <span style="color:red;">Admin</span></p>
```

Real-Life Example
 is like highlighting a word with a marker in a sentence.

Interview Notes
- Used inside text or paragraphs
- Best for inline styling

**Difference Between and **

Feature		
Type	Block	Inline
New Line	Yes	No
Width	Full width	Content width
Usage	Layout	Inline styling

Quick Interview Summary
- → grouping & layout
- → inline text styling
- Both are non-semantic elements

HTML Navigation

Definition:
Navigation in HTML is used to provide links that help users move between pages or sections of a website.

Why / Purpose
- Helps users navigate easily
- Improves user experience
- Essential for website structure and usability

Key Points

- Usually created using links (<a>)
- HTML5 provides the <nav> semantic tag
- Can be horizontal or vertical
- Often contains menus like Home, About, Contact

HTML5 <nav> Element

```
<nav>
  <a href="home.html">Home</a>
  <a href="about.html">About</a>
  <a href="contact.html">Contact</a>
</nav>
```

Navigation Using List (Best Practice)

```
<nav>
  <ul>
    <li><a href="#">Home</a></li>
    <li><a href="#">Services</a></li>
    <li><a href="#">Contact</a></li>
  </ul>
</nav>
```

Interview Notes

- <nav> is a semantic element
- Improves SEO and accessibility
- Not all links need to be inside <nav>

Quick Interview Summary

Element	Purpose
<nav>	Navigation container
<a>	Navigation link
	Menu structure

HTML Semantic Elements

Definition:
Semantic elements are HTML elements that clearly describe their meaning and purpose to both the browser and the developer.

Why / Purpose

- To make code more readable and meaningful
- To improve SEO (Search Engine Optimization)
- To enhance accessibility for screen readers

Key Points

- Introduced mainly in HTML5
- Replace excessive usage of
- Help browsers understand page structure
- Improve maintainability of code

Common Semantic Elements

Element	Purpose
<header>	Top section / page header
<nav>	Navigation links
<section>	Related content group
<article>	Independent content
<aside>	Side content
<footer>	Bottom section
<main>	Main content of page

Example

```
<header>
  <h1>My Website</h1>
</header>

<nav>
  <a href="#">Home</a>
  <a href="#">About</a>
</nav>

<main>
  <section>
    <article>
      <h2>News</h2>
      <p>HTML5 semantic elements explained</p>
    </article>
  </section>
</main>

<footer>
  <p>© 2025</p>
</footer>
```

Real-Life Example

Semantic elements are like labeled rooms in a building – kitchen, bedroom, hall – easy to understand their purpose.

Non-Semantic vs Semantic

Non-Semantic	Semantic
	<section>
	<article>
Meaning not clear	Meaning clear

Interview Notes

- Semantic tags improve SEO & accessibility
- <main> should be used only once per page
- <article> can exist independently
- <section> is for related content

Quick Interview Summary

- Semantic = meaningful tags
- HTML5 introduced semantic elements
- Improves structure, SEO, and readability

<header>

Definition:
<header> represents the introductory content or heading of a page or section.
Purpose
- Holds logo, title, headings
- Can appear multiple times

Key Points
- Semantic element
- Can contain <h1>–<h6>, <nav>
- Not a replacement for <head>

Example
```
<header>
  <h1>My Website</h1>
</header>
```
Interview Notes
- <header> ≠ <head>
- Can be inside <article> or <section>

<nav>

Definition:
<nav> defines a section that contains navigation links.
Purpose
- Helps users move across pages

Key Points
- Contains major navigation links
- Improves SEO & accessibility

Example
```
<nav>
  <a href="#">Home</a>
  <a href="#">About</a>
</nav>
```
Real-Life Example
Like a menu card.
Interview Notes
- Not all links go inside <nav>

<article>

Definition:
<article> represents independent, self-contained content.
Purpose
- Used for blogs, news, posts

Key Points
- Can be reused or shared
- Can contain its own <header> & <footer>

Example
```
<article>
  <h2>HTML5</h2>
  <p>New semantic elements introduced.</p>
```

```
</article>
```
Real-Life Example
Like a newspaper article.
Interview Notes
- Content should make sense standalone

<section>

Definition:
<section> groups related content under a theme.

Purpose
- Logical grouping of content

Key Points
- Usually has a heading
- Not independent like <article>

Example
```
<section>
  <h2>Courses</h2>
  <p>HTML, CSS, JS</p>
</section>
```
Real-Life Example
Like chapters in a book.
Interview Notes
- Use <section> when content is related

<footer>

Definition:
<footer> represents the bottom section of a page or section.
Purpose
- Holds copyright, links, contact info

Key Points
- Can appear multiple times
- Usually at the bottom

Example
```
<footer>
  <p>© 2025 MySite</p>
</footer>
```
Real-Life Example
Like end credits in a movie.
Interview Notes
- Can be inside <article> or page footer

Quick Comparison Table

Tag	Usage
<header>	Intro / heading
<nav>	Navigation links
<article>	Independent content
<section>	Related content
<footer>	Bottom info

Web Forms 2.0 (HTML5 Forms)

Definition:
Web Forms 2.0 refers to the enhanced form features introduced in HTML5, including new input types, attributes, and built-in validation.

Why / Purpose

- To collect user input efficiently
- To reduce JavaScript dependency for validation
- To improve user experience

Key Features of Web Forms 2.0

- New input types (email, date, number, etc.)
- New form attributes (required, placeholder)
- Built-in form validation
- Better mobile support

Common HTML5 Input Types

Input Type	Purpose
text	Normal text
email	Email validation
password	Hidden input
number	Numeric input
date	Date picker
tel	Phone number
url	URL validation
range	Slider
color	Color picker

Example

```
<form>
  <input type="email" placeholder="Enter email" required>
  <input type="number" min="18" max="60">
  <input type="submit">
</form>
```

New HTML5 Form Attributes

Attribute	Purpose
required	Mandatory field
placeholder	Hint text
autofocus	Focus on load
pattern	Regex validation
readonly	Read-only input
disabled	Disable input

Real-Life Example
Web Forms 2.0 is like a smart application form that checks mistakes automatically before submission.

Interview Notes

- Validation works without JavaScript

- Better for mobile keyboards
- Backward compatible
- pattern uses regular expressions

Quick Interview Summary

- Web Forms 2.0 = HTML5 forms
- Reduces JS code
- Improves UX & validation

HTML Form Elements

Definition:
Form elements are HTML elements used to collect user input and send it to a server.

Why / Purpose

- To gather data from users
- Used in login, registration, feedback forms

Main Form Elements

Element	Purpose
<form>	Form container
<input>	User input
<label>	Input label
<textarea>	Multi-line text
<select>	Dropdown
<option>	Dropdown item
<button>	Action button

Basic Form Example

```
<form>
  <label>Username:</label>
  <input type="text"><br>
  <label>Password:</label>
  <input type="password"><br>
  <button>Login</button>
</form>
```

Real-Life Example
Forms are like paper application forms.

Interview Notes

- <form> has action and method attributes
- method → GET or POST

HTML Input Types

Definition:
Input types define what kind of data the user can enter.

Why / Purpose

- Improves data accuracy
- Enables built-in validation

Common Input Types

Type	Usage
text	Normal text
password	Hidden text
email	Email validation
number	Numbers only
radio	Single selection
checkbox	Multiple selection
file	File upload
submit	Submit form
reset	Reset form
date	Date picker

Example

```
<input type="email" required>
<input type="radio" name="gender"> Male
<input type="checkbox"> Java
```

Real-Life Example
Input types are like different boxes in a form (text box, checkbox, date picker).

Interview Notes

- radio buttons must share same name
- checkbox allows multiple values
- HTML5 input types improve mobile UX

Quick Interview Summary

- Form elements collect data
- Input types control data format
- HTML5 provides built-in validation

Form Validation

Definition:
Form validation is the process of checking user input to ensure it is correct, complete, and in the required format before submitting the form.

Why / Purpose

- To prevent invalid or incomplete data
- To improve user experience

- To reduce server-side errors

Types of Form Validation

1. Client-side validation (Browser / HTML / JavaScript)
2. Server-side validation (Backend)

Client-Side Validation (HTML5)

Common Validation Attributes

Attribute	Purpose
required	Field must be filled
minlength	Minimum characters
maxlength	Maximum characters
pattern	Regex-based validation
min, max	Number/date limits
type="email"	Email validation

HTML5 Validation Example

```
<form>
  <input type="email" required placeholder="Enter email">
  <input type="password" minlength="6" required>
  <input type="submit">
</form>
```

JavaScript Validation Example

```
<script>
function validate() {
  let name = document.getElementById("name").value;
  if(name === "") {
    alert("Name is required");
    return false;
  }
}
</script>
```

Server-Side Validation

- Done after form submission
- Implemented using Java, PHP, Node.js, etc.
- More secure than client-side validation

Real-Life Example

Form validation is like a security guard checking details before entry.

Interview Notes

- HTML5 validation reduces JS code
- Client-side validation can be bypassed
- Server-side validation is mandatory

Quick Interview Summary

- Validation ensures correct input
- Two types: client-side & server-side
- HTML5 provides built-in validation

Events

Definition:
An event is an action or occurrence that happens in the browser as a result of user interaction or system activity.

Why / Purpose

- To make web pages interactive
- To respond to user actions like clicks, typing, loading, etc.

Common Types of Events

Category	Examples
Mouse Events	onclick, ondblclick, onmouseover
Keyboard Events	onkeydown, onkeyup
Form Events	onsubmit, onchange, onfocus
Window Events	onload, onresize

Event Handling Methods
Inline Event Handling

```
<button onclick="showMsg()">Click</button>
```

JavaScript Event Listener (Best Practice)

```
<script>
document.getElementById("btn")
  .addEventListener("click", function() {
    alert("Clicked");
  });
</script>
```

Example

```
<input type="text" onfocus="alert('Focused')">
```

Real-Life Example
Events are like switches – action happens only when you press them.

Interview Notes

- addEventListener() is preferred
- One event can have multiple listeners
- Inline events are not recommended

Quick Interview Summary

- Events = user actions
- Used to create interactivity
- Best practice → addEventListener()

Web Storage

Definition:
Web Storage is an HTML5 feature that allows websites to store data in the browser in the form of key–value pairs.

Why / Purpose

- To store data on the client side
- To reduce server requests
- Faster than cookies and more storage capacity

Key Points

- Data stored as key–value pairs
- Stores data in browser memory
- Data is stored as strings
- Not sent with every HTTP request (unlike cookies)

Types of Web Storage

1. Local Storage
2. Session Storage

Real-Life Example

Web Storage is like notes saved in your browser notebook.

Interview Notes

- Introduced in HTML5
- Storage limit ≈ 5–10 MB

Local Storage

Definition:

Local Storage stores data in the browser without expiration.

Why / Purpose

- To persist data even after browser is closed
- Used for user preferences, themes, tokens

Key Points

- Data persists until manually cleared
- Accessible across all tabs of same origin
- Stored as key–value pairs

Example

```
localStorage.setItem("username", "admin");
localStorage.getItem("username");
localStorage.removeItem("username");
localStorage.clear();
```

Real-Life Example

Local storage is like saving contacts in your phone — they remain even after restart.

Interview Notes

- Data never expires automatically
- Shared across tabs

Session Storage

Definition:
Session Storage stores data only for the duration of a browser session.

Why / Purpose

- To store temporary data
- Used in form steps, login sessions

Key Points

- Data is cleared when tab is closed
- Not shared across tabs
- Same API as Local Storage

Example

```
sessionStorage.setItem("otp", "1234");
sessionStorage.getItem("otp");
sessionStorage.removeItem("otp");
sessionStorage.clear();
```

Real-Life Example
Session storage is like writing on a whiteboard – erased when session ends.

Interview Notes

- Tab-specific storage
- Useful for sensitive temporary data

Local Storage vs Session Storage

Feature	Local Storage	Session Storage
Lifetime	Permanent	Until tab closes
Scope	All tabs	Single tab
Storage Size	~5–10 MB	~5 MB
Use Case	Preferences	Temporary data

Quick Interview Summary

- Web Storage = HTML5 feature
- Local Storage → permanent
- Session Storage → temporary
- Faster & larger than cookies

Web SQL Database

Definition:
Web SQL Database is a browser-based database API that allows storing and managing structured data using SQL on the client side.

Why / Purpose

- To store large, structured data in the browser
- To perform SQL queries like CREATE, INSERT, SELECT

Key Points

- Uses SQLite internally
- Works only in some browsers (Chrome, Safari)
- Deprecated (no longer recommended for new apps)
- Data stored on client side

Basic Syntax

```
var db = openDatabase(
  "myDB", "1.0", "Test DB", 2 * 1024 * 1024
);
```

Creating Table Example

```
db.transaction(function (tx) {
  tx.executeSql(
    "CREATE TABLE IF NOT EXISTS users (id INTEGER, name TEXT)"
  );
});
```

Insert Data Example

```
db.transaction(function (tx) {
  tx.executeSql(
    "INSERT INTO users VALUES (1, 'Admin')"
  );
});
```

Select Data Example

```
db.transaction(function (tx) {
  tx.executeSql(
    "SELECT * FROM users",
    [],
    function (tx, results) {
      console.log(results.rows.item(0).name);
    }
  );
});
```

Real-Life Example

Web SQL is like Excel with SQL queries inside the browser.

Interview Notes

- Web SQL is deprecated
- Not supported in Firefox & Edge
- Use IndexedDB instead

Web SQL vs Web Storage

Feature	Web SQL	Local Storage
Data Type	Structured	Key–Value
Query Support	SQL	No
Status	Deprecated	Active
Size	Large	Limited

Quick Interview Summary

- Web SQL uses SQLite
- SQL-based browser DB

- Deprecated → use IndexedDB

Geolocation API

Definition:
The Geolocation API allows a web application to get the geographical position (latitude, longitude) of a user's device.

Why / Purpose
- To provide location-based services
- Used in maps, delivery apps, weather apps
- Enhances user experience with personalized content

Key Points
- Part of HTML5 API
- Requires user permission to access location
- Can return latitude, longitude, accuracy, altitude, speed

Basic Syntax
```
navigator.geolocation.getCurrentPosition(success, error);
```
- success → callback function if location is obtained
- error → callback function if location fails

Example
```
function showPosition(position) {
  console.log("Latitude: " + position.coords.latitude);
  console.log("Longitude: " + position.coords.longitude);
}
navigator.geolocation.getCurrentPosition(showPosition);
```

Real-Life Example
Geolocation is like a GPS device in your phone, helping apps know your location.

Interview Notes
- Works only on secure context (https)
- Can track real-time location with watchPosition()
- Always handle permission denial gracefully

Optional: Real-Time Tracking
```
navigator.geolocation.watchPosition(function(position) {
  console.log("Latitude: " + position.coords.latitude);
  console.log("Longitude: " + position.coords.longitude);
});
```

Quick Interview Summary
- HTML5 API for location
- Requires user permission
- Returns coordinates & accuracy
- Use getCurrentPosition() and watchPosition()
