**1.** **What is HTML and why is it important?**

**Ans:**

**HTML (HyperText Markup Language)** is the standard markup language used to create and structure content on the **web**. It defines the **structure** of a webpage using a system of **tags** and **elements**.

* **HyperText**: Refers to the ability to link to other web pages via hyperlinks.
* **Markup Language**: Uses tags to "mark up" text, images, and other content for display in a web browser

Why is HTML Important?

**✅ Foundation of Web Pages**

HTML is the **basic building block** of all websites. Without HTML, browsers wouldn't know how to display content.

**✅ Structure and Semantics**

HTML provides a **semantic structure** to web content (headings, paragraphs, lists, tables), which helps browsers and screen readers understand the content better.

**✅ Search Engine Optimization (SEO)**

Proper HTML structure (like using <h1>, <meta>, etc.) improves how search engines index and rank websites.

**✅ Cross-Platform Compatibility**

HTML is supported by **all browsers** and **platforms**, making it a universal solution for creating web content.

**✅ Accessibility**

Well-structured HTML helps make websites accessible to users with disabilities through screen readers and assistive technologies.

**✅ Integration with Other Technologies**

HTML works hand-in-hand with:

* **CSS** for styling
* **JavaScript** for interactivity
* **APIs** and **frameworks** for dynamic web apps

**2.** **What is the difference between HTML and HTML5?**

**Ans:**

| **Aspect** | **HTML** | **HTML5** |
| --- | --- | --- |
| **Version** | Older versions (HTML 4.01, XHTML, etc.) | Latest standard (fifth revision of HTML) |
| **Doctype Declaration** | Long and complicated | Simple and short |
|  | <!DOCTYPE HTML PUBLIC...> | <!DOCTYPE html> |
| **Multimedia Support** | No native support for audio/video | <audio> and <video> tags introduced |
| **Graphics and Animation** | Requires Flash or external plugins | Supports <canvas> and <svg> for drawing and animation |
| **Form Controls** | Limited types (text, password, etc.) | New input types like email, date, range, color, etc. |
| **Semantics** | No specific semantic tags | New semantic elements like <header>, <footer>, <article> |
| **Browser Compatibility** | Requires plugins for modern features | Designed for modern browsers without plugins |
| **Offline Capability** | Not supported | Supports offline storage via Web Storage & Application Cache |
| **Mobile Friendly** | Not optimized | Designed with mobile and responsive design in mind |
| **Geolocation API** | Not available | Included in HTML5 standard |
| **Error Handling** | Strict, depends on DTD | More forgiving, better error handling in browsers |

**3.** **What is the purpose of the <!DOCTYPE> declaration?**

**Ans:**

The <!DOCTYPE> declaration (also called the **Document Type Declaration**) tells the **web browser** what version of **HTML** the page is written in. It ensures that the browser **renders the page correctly** by using the appropriate **parsing rules**.

**4.** **Explain the difference between <div> and <span>.**

**Ans:**

The <div> and <span> elements are **generic container tags** in HTML, but they serve different purposes based on how they behave and are used in structuring a webpage.

**🔷 <div> – Block-Level Element**

* **What it is**: <div> stands for “division” and is a **block-level** container.
* **Behavior**: It takes up the full width of its parent container by default, creating a line break before and after itself.
* **Usage**: Used to group large sections of content like paragraphs, images, or other block elements to apply styles or layouts using CSS.

**🔷 <span> – Inline Element**

* **What it is**: <span> is an **inline** container used for grouping small chunks of text or elements inside a block.
* **Behavior**: It does **not** create a line break and only takes as much width as its content needs.
* **Usage**: Ideal for applying styles to part of the text **within a paragraph or heading**.

**🔍 Key Differences:**

1. **Display Type**:
   * <div> is **block-level** (occupies full width, creates a new line).
   * <span> is **inline** (sits within a line, doesn’t break it).
2. **Usage Purpose**:
   * <div> is used to structure **sections** of a page.
   * <span> is used to style **text or inline elements**.
3. **Typical Use Case**:
   * Use <div> when you're dealing with **layout** and larger blocks.
   * Use <span> when you want to change style **within text**.

**5.** **What is semantic HTML? Give examples.**

**Ans:**

**Semantic HTML** refers to using HTML **tags that clearly describe the meaning** and **purpose of the content** they contain. These tags not only define how elements look but also convey the structure and meaning of the content to both **browsers** and **developers**, and improve **accessibility** for screen readers and **SEO** for search engines.

🧱 Examples of Semantic HTML Elements:

| **Tag** | **Purpose** |
| --- | --- |
| <header> | Defines the top section of a page or section (e.g., logo, nav). |
| <nav> | Specifies a navigation menu or list of links. |
| <main> | Represents the main content area of the document. |
| <section> | Represents a generic standalone section of content. |
| <article> | Used for independent, self-contained content (e.g., blog post, news). |
| <aside> | Contains side content like ads, sidebars, or related links. |
| <footer> | Defines the bottom section, often with contact info, copyright. |
| <figure> | Wraps content like images or charts with optional captions. |
| <figcaption> | Provides a caption for a <figure>. |
| <time> | Represents a specific time or date. |
| <mark> | Highlights text that is relevant or important. |

**6.** **What are the different types of lists in HTML?**

**Ans:**

HTML provides three main types of lists to organize content in a structured and readable way. Each type serves a specific purpose depending on how the content should be displayed or interpreted.

**🔷 1. Ordered List (<ol>)**

An **ordered list** displays items in a **specific sequence**, usually numbered (1, 2, 3...) or lettered (a, b, c...).

<ol>

<li>Wake up</li>

<li>Brush teeth</li>

<li>Eat breakfast</li>

</ol>

**🔷 2. Unordered List (<ul>)**

An **unordered list** displays items with **bullet points**, regardless of order.

<ul>

<li>HTML</li>

<li>CSS</li>

<li>JavaScript</li>

</ul>

**🔷 3. Description List (<dl>)**

A **description list** is used to display terms and their corresponding **descriptions**. It uses three elements:

* <dl> – Defines the list
* <dt> – Defines the term
* <dd> – Defines the description

<dl>

<dt>HTML</dt>

<dd>HyperText Markup Language</dd>

<dt>CSS</dt>

<dd>Cascading Style Sheets</dd>

</dl>

**7.** **What is the purpose of the alt attribute in <img> tag?**

**Ans:**

The alt attribute (short for **alternative text**) in the <img> tag serves several **important purposes** related to **accessibility**, **SEO**, and **user experience**.

If an image **fails to load** due to a broken link or slow internet, the browser will show the alt text in place of the image.

Search engines **cannot see images** but they can **read alt text**.

Providing meaningful alt descriptions helps search engines **index images properly**, making your content more searchable.

**8.** **What is the difference between id and class?**

**Ans:**

Both id and class are **HTML attributes** used to **identify and style elements**, but they have different purposes, rules, and usage patterns.

**🔷 1. id Attribute**

* **Uniquely identifies a single element** on the page.
* Should be **used only once** per page.
* Often used for JavaScript targeting or anchor navigation.
* Selector in CSS: uses a **hash #**.

**🔷 2. class Attribute**

* Used to **group multiple elements** with the same style or behavior.
* Can be used **many times** across the page.
* An element can have **multiple classes**.
* Selector in CSS: uses a **dot .**.

| **Feature** | **id** | **class** |
| --- | --- | --- |
| Usage Limit | Only **once** per page | Can be used on **multiple** elements |
| Purpose | Unique identifier | Reusable grouping |
| CSS Selector | #idname | .classname |
| JavaScript Targeting | Ideal for targeting a single element | Good for groups of elements |
| Multiplicity | One id per element | Can assign **multiple classes** |
| Specificity in CSS | Higher specificity (overrides class) | Lower specificity |

**9.** **What are void (self-closing) elements in HTML?**

**Ans:**

**Void elements** (also called **self-closing tags**) are **HTML elements that do not have closing tags** because they **cannot contain any content**. They are used to insert standalone elements like images, line breaks, or input fields.

**🔍 Characteristics of Void Elements:**

1. ✅ **They don’t have closing tags** (no </tag>).
2. ✅ **They don’t contain child elements or text.**
3. ✅ **They are syntactically complete with just one tag.**
4. ✅ Commonly used for inserting **media**, **form controls**, **spacing**, etc.

| **Tag** | **Purpose** |
| --- | --- |
| <br> | Inserts a line break |
| <hr> | Adds a horizontal line (thematic break) |
| <img> | Embeds an image |
| <input> | Creates a form input field |
| <meta> | Defines metadata inside <head> |
| <link> | Links external resources like CSS files |
| <area> | Defines a clickable area inside an image map |
| <base> | Sets the base URL for all relative links |
| <col> | Specifies column properties in a table |
| <embed> | Embeds external content like videos or PDFs |
| <source> | Specifies source for <audio> or <video> |
| <track> | Adds subtitles/captions for media content |
| <wbr> | Suggests a line break opportunity in long words |

**10.** **What are the new features introduced in HTML5?**

**Ans:**

HTML5 introduced many **modern, powerful features** to make web development more efficient, interactive, and user-friendly. These enhancements improved **semantics**, **media handling**, **graphics**, **forms**, **storage**, and **performance** — making HTML5 ideal for building dynamic, mobile-ready web applications.

**🔷 1. New Semantic Elements**

HTML5 added new **semantic tags** to give meaningful structure to web pages. These improve **accessibility**, **SEO**, and **code readability**.

<header>, <footer>, <section>, <article>, <nav>, <aside>, <main>, <figure>, <figcaption>

**🔷 2. Multimedia Support Without Plugins**

HTML5 supports **audio and video** directly in the browser without needing Flash or other external plugins.

* <audio> – Embeds audio files
* <video> – Embeds video files

**🔷 3. New Form Input Types & Attributes**

HTML5 introduced many new **input types** and **attributes** to improve form usability and validation:

email, date, time, url, range, color, number, search, tel

**🔷 4. Graphics and Drawing**

HTML5 added support for drawing and graphics directly in the browser.

* <canvas> – Used for 2D graphics, animations, games, data visualizations
* <svg> – Scalable Vector Graphics for resolution-independent graphics

**11.** **How can you embed multimedia in HTML?**

**Ans:**

You can embed **multimedia** such as **audio**, **video**, and **interactive content** directly into HTML pages using modern tags introduced in **HTML5**. These allow you to add rich media without needing external plugins like Flash.

**🔷 1. Embedding Video with <video> Tag**

The <video> element allows you to embed video files with built-in controls like play, pause, and volume.

**🔷 2. Embedding Audio with <audio> Tag**

The <audio> element allows you to play sound files like music, narration, or effects.

**🔷 3. Embedding YouTube Videos (or Other Media) with <iframe>**

The <iframe> tag is used to embed external content like YouTube videos, Google Maps, or other websites.

**🔷 4. Using <embed> and <object> (For PDFs, Flash, or Legacy Media)**

These tags are more flexible but less common today due to security concerns and lack of mobile support.

**12.** **Difference between <section>, <article>, <aside>, and <nav>?**

**Ans:**

These tags are **semantic HTML5 elements** that provide **meaningful structure** to your web pages. They help browsers, developers, screen readers, and search engines **understand the purpose** of each part of a webpage.

| **Tag** | **Purpose** | **Typical Use** |
| --- | --- | --- |
| <section> | Thematic grouping of related content | About, Services, Features sections |
| <article> | Independent, reusable content | Blog posts, news articles, reviews |
| <aside> | Supplementary or side content | Sidebars, ads, related links, tips |
| <nav> | Navigation section for internal/external links | Menus, breadcrumbs, table of contents |

**CSS (Cascading Style Sheet):**

1. What is the Difference Between Inline, Internal, and External CSS?

Ans:

| **Type** | **Where It's Written** | **Scope & Use Case** |
| --- | --- | --- |
| **Inline** | Inside an element’s style attribute | Affects only that element; used for quick fixes |
| **Internal** | Inside a <style> tag in <head> | Affects the whole page; best for single-document sites |
| **External** | In a separate .css file linked via <link> | Reusable across multiple pages; best practice for large sites |

2. What is the Box Model in CSS?

Ans:

The **CSS Box Model** describes how every HTML element is structured in rectangular boxes:

1. **Content** – The actual text or image.
2. **Padding** – Space around the content (inside the element).
3. **Border** – Line surrounding the padding and content.
4. **Margin** – Space outside the border (separates elements).

3. What is the Difference Between em, rem, px, and % Units?

Ans:

| **Unit** | **Based On** | **Use Case** |
| --- | --- | --- |
| px | Absolute pixels | Fixed-size elements |
| em | Relative to parent element’s font size | Scalable, but can be hard to manage nesting |
| rem | Relative to root (<html>) font size | Scalable and consistent |
| % | Relative to parent element’s size | Useful for responsive widths, heights |