

Website & Its Fundamentals

— A Beginner's Guide to the World of Websites

The internet has become the backbone of modern communication, business, education, and daily life—and at the heart of it all lies the **website**. Whether you're searching for information, shopping online, accessing services, or building your online presence, websites play an essential role in how we interact with the digital world.

This book, “**Website & Its Fundamentals**,” is thoughtfully designed as an introductory guide for students, beginners, and curious learners who want to understand what websites are and how they work. With simple explanations, visual examples, and real-world references, this book lays the groundwork for anyone interested in web development or digital literacy.

Key Features of This Book:

- **Basic to Advanced Coverage:** The book starts from the very basics—what a website is, types of websites, domain names, and hosting—and gradually introduces core technologies like **HTML, CSS, and JavaScript** in a structured and beginner-friendly manner.
- **Step-by-Step Learning:** Concepts are explained using **simple language, visual diagrams, and example-based learning** to ensure a smooth experience even for absolute beginners.
- **Practical Examples:** Real-world website scenarios, basic web page creation, and common use cases are included to help readers relate the theory with practical application.
- **Interactive Activities:** Each chapter includes **exercises, questions, and hands-on tasks** that encourage learners to build and explore their own simple web pages as they learn.

Who Should Read This Book?

This book is ideal for:

- School and college students exploring the basics of the internet and websites.
- Beginners with no prior coding experience.
- Entrepreneurs or freelancers wanting to create their own websites.
- Anyone curious about how websites are built and function behind the scenes.

It is a stepping stone for more advanced studies in **web development, UI/UX design, or digital marketing**.

Acknowledgments

This book is the result of countless hours of research, writing, and feedback from educators, students, and tech professionals. I sincerely thank everyone who supported me during this journey. A special thanks to my readers—you inspire me to continue simplifying learning for all.

Conclusion

In this digital age, learning how websites work is more than a skill—it's a gateway to countless opportunities. Through this book, I hope to equip you with the confidence and clarity needed to take your first steps into the world of websites. May this journey be both educational and exciting for you.

I hope you find this book both informative and inspiring as you embark on your journey into the fascinating world of Website.

Happy Learning!



Author
Sunil Kumar

Contents

What is WWW?	1
Difference between Internet and WWW:.....	2
Functions of WWW Components:	2
 Browser:.....	3
 History of Web Browsers	4
 Search Engine	4
Types of Search Engines: -	5
 History:	5
How does search engine works?	6
1. Crawling – The Foundation of Search Engine Discovery.....	6
List of  Search Engine: -	7
Website?	8
 Website: Definition & Types	8
 Types of Websites.....	8
How the Web Works (Client–Server Model)	9
 Web Server: Definition, Use, and Examples.....	11
Working of Web Server:	12
WEB PAGE	13
 Web Page – Key Elements & Features	13
 Types of Webs Pages	14
HTML	16
HYPER TEXT MARKUP LANGUAGE	16
 Features of HTML (HyperText Markup Language).....	16
Structure of an HTML Page.....	19
Important HTML Head Tags & Their Functions.....	20
 Building Blocks of HTML.....	24
◆ 1. Tags.....	24
◆ 2. Attributes.....	24
◆ 3. Elements.....	25
To see the HTML code for any website, follow these methods:.....	26
◆ 1. Visual Studio Code (VS Code)	27
◆ 2. Sublime Text.....	28
◆ 3. Atom	28
◆ 4. Notepad++.....	28
◆ 5. Brackets.....	28

◆ 6. Adobe Dreamweaver	29
◆ 7. Emacs.....	29
◆ 8. Vim.....	29
◆ 9. CodePen (Online Editor)	30
◆ 10. JSFiddle (Online Editor).....	30
◆ 11. Replit (Repl.it)	30
✓ How to Write HTML Code in Visual Studio Code	31
◆ 1. Common Attributes (सामान्य एट्रिब्यूट्स)	39
✓ HTML Elements –	43
◆ Types of HTML Elements:.....	43
◆ Block-Level Elements.....	45
◆ HTML HEADING TAGS	47
◆ HTML PARAGRAPH TAGS	47
◆ HTML TEXT FORMATTING	49
◆ HTML QUOTATION AND CITATION ELEMENTS	51
◆ HTML PHRASE TAG	52
◆ HTML COMMENT TAG	53
◆ HTML STYLES	54

What is WWW?

WWW stands for **World Wide Web**, and it is **not the internet itself**, but rather a **service** that runs **on top of the internet** to make it easy for users to access, share, and interact with information through **webpages and websites** using **browsers** like Chrome, Firefox, or Safari.

Definition of WWW: The **World Wide Web (WWW)** is a system of **interlinked hypertext documents** accessed via the Internet. Users can view web pages that may contain text, images, videos, and other multimedia, and navigate between them using hyperlinks.

www is a global collection of documents and other resources linked by hyperlink and URLs. it is known as web, it is an information system technology enabling.

History of the World Wide Web (WWW)

Year	Event
1989	Tim Berners-Lee , a British scientist at CERN (European Organization for Nuclear Research), proposed the idea of the World Wide Web as a way to share information globally among scientists.
1990	He developed the first web browser (called Worldwide Web , later renamed Nexus) and the first web server (http://info.cern.ch).
1991	The World Wide Web was released publicly for use across the Internet. The first website (info.cern.ch) went live.
1993	Mosaic , the first graphical web browser , was developed by Marc Andreessen . It helped the Web gain popularity.
1994	Netscape Navigator was launched – a more advanced web browser. Also, W3C (World Wide Web Consortium) was founded by Tim Berners-Lee to standardize web technologies.
1995	The first major websites appeared – Amazon, eBay, Yahoo . The web started supporting JavaScript and multimedia .
1998	Google was founded, revolutionizing web search and becoming the most powerful search engine.
2004	The rise of Web 2.0 : websites became interactive (social media, blogs, forums). Facebook and YouTube emerged.
2010–Present	The web became mobile-friendly . Web apps, cloud computing, e-learning, and e-commerce saw massive growth.

Important key to keep in mind:

📌 **First website:** <http://info.cern.ch>

📌 **First browser:** WorldWideWeb/Nexus

📌 **First graphical browser:** Mosaic

📌 **First search engine:** Archie (pre-Web), later Yahoo and Google

“The **WWW was invented in 1989** by **Tim Berners-Lee** at CERN to share scientific information. Over time, it evolved into a global platform for **communication, information sharing, commerce, education, and entertainment.**”

Features of WWW:

- Uses **hypertext and hyperlinks**.
- **Platform-independent** (accessible on any device with a browser).
- Supports **multimedia** content (text, images, audio, video).
- Uses **client-server architecture**.
- Enables **interactivity** (forms, e-commerce, social media).

Real-World Examples:

- Visiting <https://www.wikipedia.org> to read articles.
- Using **YouTube** to watch videos (a multimedia web service).
- Shopping on **Amazon** through an interactive web interface.

Difference between Internet and WWW:

Internet	World Wide Web (WWW)
Network of networks	Collection of information on the internet
Hardware-based infrastructure	Software system (runs on top of internet)
Includes emails, FTP, VoIP, etc.	Only deals with web pages and resources
Existed before WWW	Invented in 1989, launched in 1991

Functions of WWW Components:

Function: - 1).HTML 2). Linking 3). www prefix 4). Scheme specifiers 5). Web Page 6). Website 7). Browser 8). Search Engine 9). Server 10). Cookie 11). Deep web 12). Caching 13). Security 14). Privacy 15). Standards

S.No.	Component	Function / Role
1.	HTML (HyperText Markup Language)	The foundation language for creating and structuring web pages. It defines elements like text, images, links, forms, and layout. Hypertext Markup Language it used for Creating Web page & Web Application.
2.	Linking (Hyperlinking)	Connects different web pages or documents via clickable text or images , allowing smooth navigation across the web . / It is interconnecting the web page via Hyperlinks .
3.	WWW Prefix	Refers to the " World Wide Web ". It is used in URLs to signify a web-based service, though it's mostly optional in modern domains. It is like .com, .org, .net
4.	Scheme Specifiers (HTTP/HTTPS)	Indicates the protocol to be used when accessing a resource. HTTP is for standard access; HTTPS is the secure encrypted version. (http:// or https://)

5.	Web Page	A single HTML document viewed in a web browser. It can include text, multimedia, and links to other pages. A webpage is an HTML document on the WWW.
6.	Website	A collection of related web pages hosted under a domain name (e.g., www.example.com). Websites may have multiple sections and purposes.
7.	Browser	A software application (e.g., Chrome, Firefox) used to retrieve, interpret, and display content from the web. It is a software responsible for open the website.
8.	Search Engine	A tool like Google that indexes websites and helps users find information quickly using keyword searches.
9.	Server (Web Server)	A computer system that stores website files and delivers them to clients (browsers) upon request.
10.	Cookie	A small data file saved on a user's browser to store preferences, login status, or track activity across sessions. It is a small piece of data sent from the website and stored on the user's computer by the web browser while user is browsing. It is useful.
11.	Deep Web	The part of the web not indexed by search engines , including password-protected sites, private databases, and internal portals.
12.	Caching	The temporary storage of web content to improve load times and reduce server load. Used by browsers and proxy servers. A web cache is a server computer located on the public internet. It is stores recently accessed web page to improve response time for user's.
13.	Security	Ensures that data transmission is protected from unauthorized access , using methods like HTTPS, encryption, firewalls , etc.
14.	Privacy	Refers to the protection of user data , such as browsing history, personal info, and location, from misuse or tracking.
15.	Standards	Defined by organizations like W3C , these ensure uniformity in how web technologies (HTML, CSS, JavaScript) work across all browsers and platforms.

Browser:

Definition

A **browser** is an **application software** (or software program) that allows users to **access, retrieve, and view** content on the **World Wide Web (WWW)** or from **local storage**. It interprets and displays HTML files and supports various web technologies.

Use

- A browser is used to **visit websites, fetch content** from the **WWW or local files**, and **display webpages** (text, images, videos, animations, etc.) on the user's device.
- It acts as an interface between the **user** and the **web server**.

History of Web Browsers

Year	Browser	Developer / Note
1990	Worldwide Web (later renamed Nexus)	First browser, by Sir Tim Berners-Lee
1993	Mosaic	First graphical browser
1994	Netscape Navigator	Very popular in early web days
1995	Internet Explorer	Developed by Microsoft
1995	Opera	Lightweight browser from Norway
2003	Safari	Apple's web browser
2004	Mozilla Firefox	Open-source browser with focus on privacy
2008	Google Chrome	Fast, secure, and popular browser from Google
2015	Microsoft Edge	Successor of Internet Explorer with modern features


Key Features of Modern Browsers

- Automatically **log browsing history**
- **Bookmarks** for saving favorite pages
- **Browser extensions** and add-ons for customization
- **Password management**
- **Sync service** across multiple devices
- **Web accessibility** support
- **Open multiple tabs/windows**
- **Back and forward buttons**
- **Refresh / Reload / Stop / Home** buttons
- **Address bar** for entering URLs (IP or domain)
- Support for **secure browsing** (HTTPS)
- **Private/Incognito mode** for anonymous browsing

Search Engine

Definition:

A **search engine** is a **software system** or a **set of programs** designed to **search the World Wide Web** for specific information based on **textual, voice, or multimedia queries**.

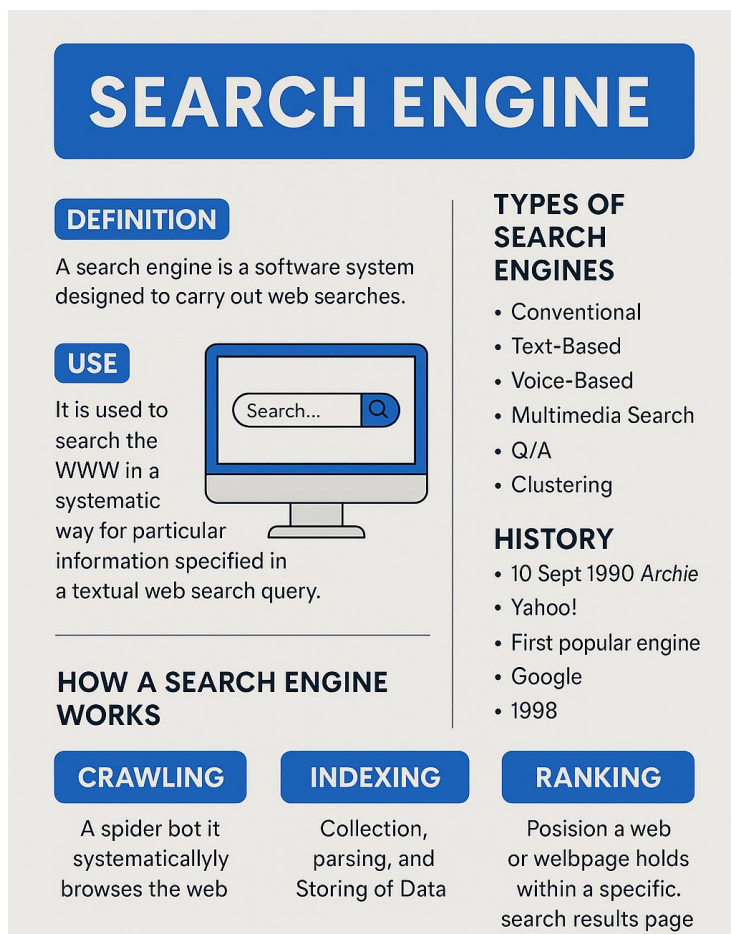
 **Use:** It is used to **search and retrieve information** from the WWW in a **systematic and organized way** based on keywords or phrases provided by the user.

Types of Search Engines: -

Type	Description
1. Conventional	Like library systems, searching by title, author, etc.
2. Text-Based	Users enter keywords to search (e.g., Google, Bing)
3. Voice-Based	Users speak queries (e.g., Google Assistant, Siri)
4. Multimedia Search	Searches by image, video, color, shape (e.g., QBIC, WebSeek)
5. Q/A (Question/Answer)	Supports natural language queries (e.g., StackExchange)
6. Clustering	Groups search results into categories (e.g., Vivisimo, Clusty)
7. Research System	For academic/research purposes (e.g., Lemur, Nutch)

History:

Year	Event
1990	First search engine "Archie" developed by Alan Emtage on 10 September 1990 . It indexed FTP sites (not full text).
1994	Yahoo! was the first popular search engine, founded by Jerry Yang and David Filo .
1998	Google was launched by Larry Page and Sergey Brin , introducing PageRank Algorithm, indexing , and hyperlink analysis to rank search results.



How does search engine works?

Search engines work through a three-step process: crawling, indexing, and ranking. First, crawlers (also known as spiders or bots) discover and scan web pages. Next, the content of these pages is analyzed and stored in a massive database called the index. Finally, when a user enters a search query, the search engine retrieves and ranks the most relevant results from the index. (खोज इंजन तीन-चरणीय प्रक्रिया के माध्यम से काम करते हैं: क्रॉलिंग, इंडेक्सिंग और रैंकिंग। सबसे पहले, क्रॉलर (जिन्हें स्पाइडर या बॉट भी कहा जाता है) वेब पेजों को खोजते हैं और स्कैन करते हैं। इसके बाद, इन पृष्ठों की विषय-वस्तु का विश्लेषण किया जाता है और उसे इंडेक्स नामक एक विशाल डाटाबेस में संग्रहीत किया जाता है। अंततः, जब कोई उपयोगकर्ता कोई खोज क्वेरी दर्ज करता है, तो खोज इंजन सूचकांक से सबसे अधिक प्रासंगिक परिणाम प्राप्त करता है और उन्हें रैंक करता है।)

1. Crawling – The Foundation of Search Engine Discovery

What is Crawling?

Crawling is the *first step* in the search engine process. It involves automated bots (also known as crawlers or spiders) that systematically **scan the web** to discover publicly available webpages. These bots:

- Visit websites
- Read HTML code
- Understand page structure
- Identify content type and meaning
- Check the creation or last update time

2. Indexing – Organizing the Web

What is Indexing?

After crawling, the data collected is **organized and stored** in massive databases known as **search engine indexes**. These indexes help the search engine **quickly retrieve relevant results**.

Search engines don't store everything. Instead, they keep:

- Page title and meta description
- Content type (text, image, video, etc.)
- Keywords associated with the content
- Internal and external links
- Additional algorithm-relevant data

3. Ranking – The Search Engine Battle

What is Ranking?

Ranking is the **position** where your website appears on a **Search Engine Results Page (SERP)**. It determines how visible your page is for a user query.

How Ranking Works: 3 Steps

Step 1: Analyze the User Query

- Search engines **break down** the user's input into meaningful **keywords**.
- They understand **user intent**, recognize **synonyms** (e.g., *change* = *replace*), and even correct **spelling mistakes**.

- Example: “how to make a chocolate cupcake” will yield **recipe-focused** results.

Step 2: Find Matching Pages

- The engine searches its **index** for pages that **best match** the intent of the query.
- Example: A query like “dark wallpaper” will return **image results**, not text-based articles, because the intent is visual.

Step 3: Display the Results

- Users are shown a **Search Engine Results Page (SERP)** with:
 - ✓ Up to **10 organic results**
 - ✓ Additional content like:
 - **Paid advertisements**
 - **Featured snippets**
 - **Direct answers**
 - **Image or video carousels**

List of Search Engine: -

S.No.	Search Engine	Founded / Established	Launch Date	Integration with Other Services
1	Google	September 1998	1998	Gmail, Google Drive, Maps, YouTube, Google Ads
2	Bing	2009	3 June 2009	Microsoft services (Outlook, Office, OneDrive)
3	Yahoo!	March 1995	1995	Yahoo Mail, Yahoo Finance, Yahoo News
4	YouTube (Search feature)	February 2005	2005	Google Search, Google Ads
5	DuckDuckGo	February 2008	25 September 2008	Limited integration (privacy-focused)
6	AOL	1985	1993 (Search feature)	AOL Mail, AOL News, AOL Video
7	Ecosia	2009	7 December 2009	Ecosia Browser Extension, Bing-powered
8	Qwant	2011	July 2013	Limited integration, privacy-focused
9	Ask.com	1996	1997	Limited services, Q&A-based
10	Dogpile	1996	1997	Aggregates multiple sources
11	Yandex	1997	23 September 1997	Yandex Mail, Disk (cloud), Maps, Translate
12	Startpage	1998	2006 (as Startpage)	Google-powered search, private integration
13	Swisscows	2014	2014	Family-friendly, private, limited integration
14	Brave Search	2021	March 2021	Brave browser, built-in privacy tools
15	You.com	2020	9 November 2021	AI chat tools, productivity apps
16	Perplexity AI	2022	2022	AI-powered Q&A, chatbots, research tools
17	Internet Archive (Search Feature)	1996	2001 (Wayback Machine)	Archive.org services, digital library
18	iAsk	~2023	2023	AI-driven, limited integration

Website?



Website: Definition & Types



Definition

A **website** is a **collection of web pages** and related content that is:

- **Accessed through the internet**
- **Identified by a common domain name**

Example: www.google.com, www.wikipedia.org



Types of Websites

Websites can be broadly classified into two main types:



1 Static Website

A **static website** contains fixed content written in HTML. Each page is a separate HTML file.

Key Features:

- Pre-built and **does not change** unless manually edited.
- Simple, **fast to load**, and **easy to host**
- No interaction with databases or user input
- Suitable for **small websites**, **portfolio pages**, or **company profiles**

Example:

A company's informational page with services and contact details.



2 Dynamic Website

A **dynamic website** can **customize content automatically** based on user behavior, time, or data from a database.

Key Features:

- Content changes **dynamically and frequently**
- Uses **server-side scripting languages** like PHP, ASP.NET, or Node.js
- Connects to **databases** (e.g., MySQL, MongoDB)
- Allows user interaction (login, comments, forms, etc.)

Example:

E-commerce websites, blogs, social media platforms (e.g., Amazon, Facebook)

Features of Statics & Dynamic Website

Feature	Static Website	Dynamic Website
Content Update	Manual	Automatic / Dynamic
Programming	Basic HTML/CSS	HTML + Server-side scripting (PHP, etc.)
Database Support	✗ No	✓ Yes
Speed	🚀 Fast	⚙️ Slightly slower due to processing
Best For	Small sites, portfolios	Large, interactive, data-driven websites

◆ Web

- The **Web** is a global system of connected computer networks that uses the Internet to access and share information. It allows users to view and exchange data through browsers. It can also be described as a **virtual directory** on a web server. In simple terms, the Web is a **part of the Internet** used to view websites and web pages.

◆ Site (Website)

- A **site** refers to a **specific location on the Web** that contains a group of related web pages. Each site is identified by a **unique domain name** and is **hosted on a web server**. You can access a site using a **URL** (Uniform Resource Locator) in a web browser. For example, www.google.com is a website.

◆ Page

- A **page** is a **single document** available on the Web. Each page can be accessed using its own **unique URL**. Pages are **part of a website** and can contain text, images, or links. For example, the "**About Us**" section of a website is a page.

◆ Web Page

- A **web page** is a **single hypertext document** available on the **World Wide Web**. It often contains text, images, videos, links, and other content. The term "hyper" means "beyond", which indicates that a web page connects to other pages or information. Web pages are written in **HTML** and are designed to be viewed in a web browser.

How the Web Works (Client–Server Model)

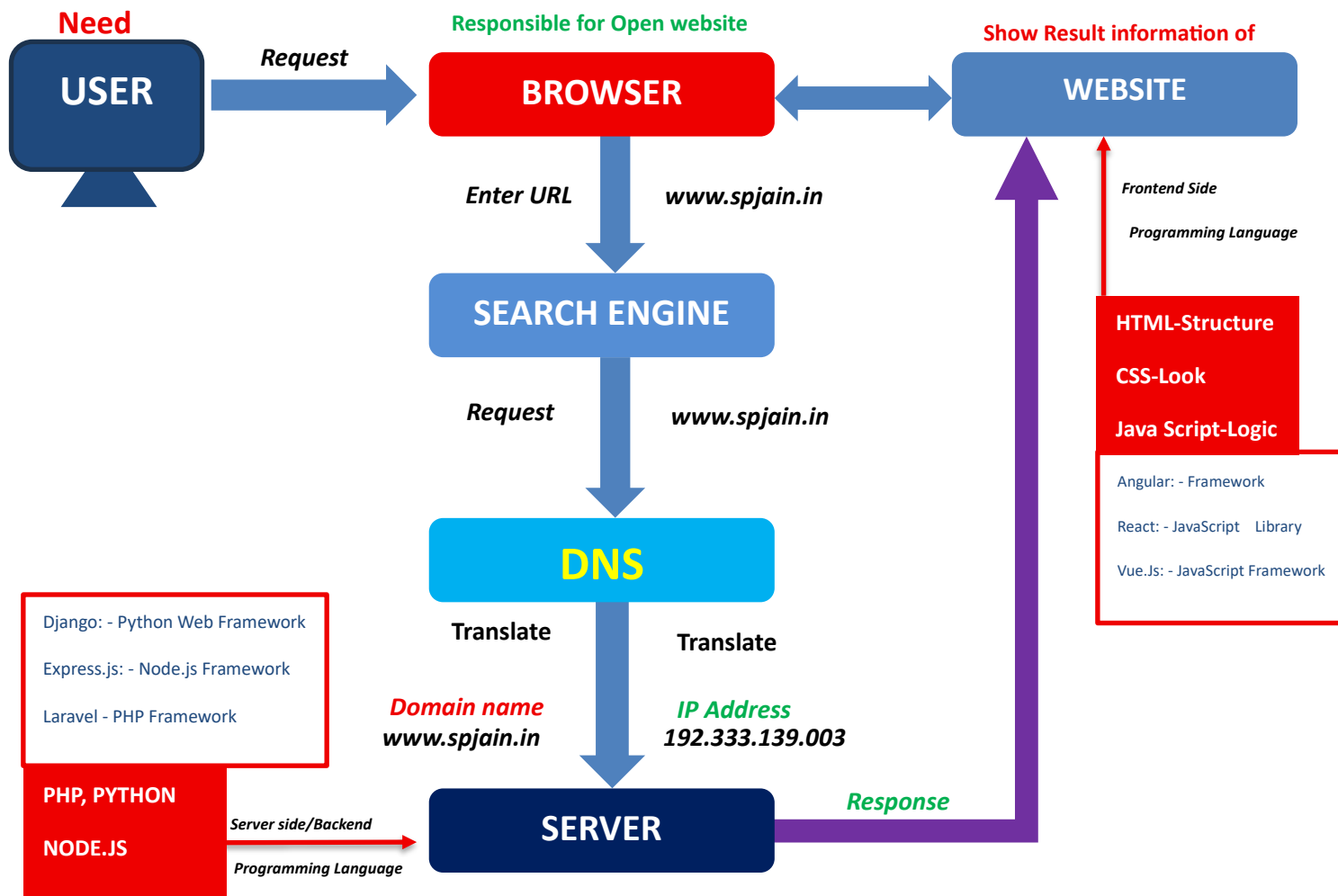
The **Web works on a client-server model**, where the **client (user's browser)** requests information from a **server**, and the server responds by sending back the requested data.

The web works through a client-server model, where users (clients) access information stored on servers. When a user enters a website address, their browser sends a request to the server hosting that website. The server retrieves the requested data (like HTML, CSS, images, etc.) and sends it back to the browser, which then renders the webpage for the user.

Here's a more detailed breakdown:

1. **Requesting a website:** A user types a website address (URL) into their browser. The browser then sends a request to a DNS server to find the corresponding IP address of the web server hosting that website.
2. **Connecting to the Server:** The browser uses the IP address to locate and connect to the web server.

3. **Sending an HTTP Request:** The browser sends an HTTP request to the server, asking for the website's files (HTML, CSS, JavaScript, images, etc.).
4. **Server Responds:** The web server receives the request and sends back the requested files.
5. **Rendering the Page:** The browser receives the files and begins to render the webpage. It parses the HTML, CSS, and JavaScript to display the content and layout to the user.
6. **Additional Requests:** The browser might make additional requests for other resources embedded in the page, like images, videos, or scripts, and repeat steps 4 and 5 for each resource.
7. **Displaying the Page:** Finally, the browser displays the fully rendered webpage for the user to interact with.
























Browser

A browser, or web browser, is a software application that allows users to access and view information on the World Wide Web. It acts as an interface between the user and the internet, enabling them to navigate, retrieve, and display web pages, images, videos, and other content.

Web browsers

From sources across the web

 Google Chrome Freeware	 Firefox GNU General Public License	 Safari Freeware
 Internet Explorer Proprietary software	 Microsoft Edge Proprietary software	 Opera Freeware
 Brave Mozilla Public License	 WorldWideWeb	 Netscape Navigator Freeware
 Google Chrome for Android BSD licenses	 Vivaldi Freeware	 Naver Whale Freeware
 UC Browser Freeware	 Yandex Browser Freeware	 Tor Browser GNU General Public License
 Opera Mini Freeware	 NCSA Mosaic Proprietary software	 Arc Proprietary software
 Zen Browser Mozilla Public License	 Chromium BSD licenses	 Opera GX Freeware

Web Server: Definition, Use, and Examples

Definition

A **Web Server** is both **hardware and software** that:

- Accepts **requests via HTTP or HTTPS protocols**
 - **Distributes web content** to users (clients)
 - Is a **dedicated computer** responsible for **running websites**
- ✓ In simple terms, a web server delivers web pages to your browser when you visit a website.

Use / Function

A web server performs the following key functions:

- **Processes HTTP/HTTPS requests** sent from the client (browser)
 - **Manages responses**, such as sending back HTML pages, images, or other files
 - **Stores website data** like HTML, CSS, images, and scripts
 - **Protects website data** using security measures (e.g., firewalls, SSL)
 - Ensures the website is always **available and accessible**
- ✓ When you click on a link, your browser sends a request, and the **web server responds** with the correct content.

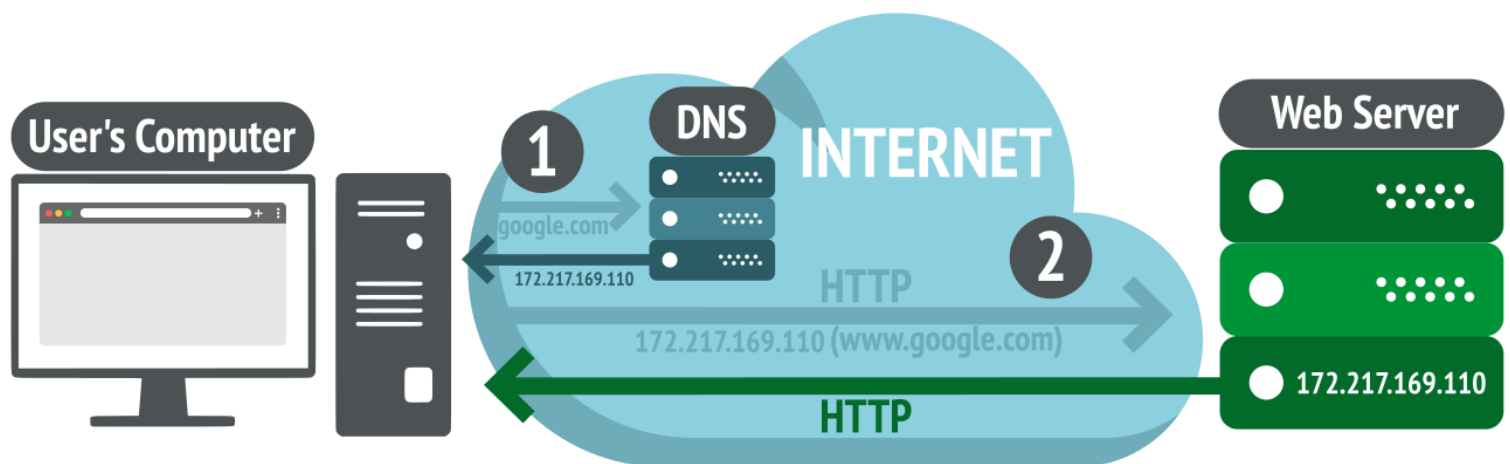
Examples of Popular Web Servers

Web Server	Description
Apache HTTP Server	One of the most widely used open-source web servers
Nginx	Lightweight and efficient, great for high-traffic websites
Microsoft IIS	Web server developed by Microsoft for Windows servers
Apache Tomcat	Specialized for Java-based web applications

Working of Web Server: -

1. **Receive Client Request/Response** (Read & Verify, URL-Normalization, URL Mapping, URL Path Redirections)
2. **Executes or refuse HTTP Request Method** (URL Authorization, URL Redirection, Directory Index File Regular Files)
3. **Response/Replies** (HTTP Response, Logs)

“A web server works by receiving requests from clients (like web browsers) and sending back responses, typically in the form of web pages and associated files. This process is facilitated by protocols like HTTP.”



❖ Steps: -

1. Client Request:

A user types a website address (URL) into their web browser, which then sends an HTTP request to the web server hosting that website.

2. DNS Resolution:

The browser needs to find the server's IP address. It does this by consulting a Domain Name System (DNS) server, which translates the domain name (e.g., example.com) into an IP address.

3. Server Connection:

The browser then uses the IP address to establish a connection with the web server.

4. Request Processing:

The web server receives the request and processes it. This might involve locating specific files (like HTML, CSS, JavaScript, images) or interacting with other systems (like databases).

5. Response:

The web server sends back the requested data to the browser. This data includes the files needed to display the webpage.

6. Page Rendering:

The browser receives the data and uses it to render and display the web page to the user.

7. Error Handling:

If the web server encounters an error (e.g., the requested page doesn't exist), it sends back an appropriate error message, such as a 404 Not Found error.

web server software companies and release dates:

- 1) **Apache HTTP Server (Apache):** Released in 1995
- 2) **Nginx:** First released in 2004
- 3) **Microsoft Internet Information Services (IIS):** Initial release in 1995
- 4) **LiteSpeed Web Server:** First release in 2003
- 5) **Google Web Server (GWS):** Private server developed by Google (Google Web Server (GWS) is Google's proprietary web server software, used internally for its web infrastructure and not released to the public.)
- 6) **Caddy:** Initial release in 2015
- 7) **IBM HTTP Server (IHS):** First released in the late 1990s
- 8) **LiteWebServer:** Initial release in 2016
- 9) **Cherokee:** First released in 2005
- 10) **Hiawatha:** Initial release in 2002

WEB PAGE

✓ **Definition:**

A **Web Page** is a **document** on the **World Wide Web (WWW)** that is written in **HTML (HyperText Markup Language)** and can be viewed in a **web browser**.



Web Page – Key Elements & Features

1 HTML (HyperText Markup Language)

- Web pages are primarily created using **HTML**, which defines the **structure and content** (like headings, paragraphs, images, links, etc.) of the page.

2 URL (Uniform Resource Locator)

- Every web page has a **unique address** called a **URL**, which allows users to **access** it via a browser.

✓ Example: <https://www.example.com/about.html>

3 Interactivity

- **JavaScript** and other technologies are used to add **interactive elements**, such as:
 - Buttons
 - Forms
 - Sliders
 - Games
 - Dynamic content (like popups or dropdowns)

4 Navigation

- **Hyperlinks (links)** connect one page to another, allowing users to:
 - Move within the same site (internal links)
 - Move to another website (external links)

5 Metadata

- Metadata gives information **about the page** but not **on the page**, like:
 - Title (<title>)
 - Description (<meta name="description">)
 - Keywords
 - Author info (for SEO and browser use)

6 Rendering

- When a user **requests a web page**:
 - The **web server sends** the HTML, CSS, and JavaScript code.
 - The **browser interprets** this code and **renders** it as a visible, usable web page.

7 Design and Styling (CSS)

- Web pages use **CSS (Cascading Style Sheets)** for:
 - Colors
 - Fonts
 - Layout
 - Responsive design (mobile-friendly display)



Types of Webs Pages

There are **two main types** of web pages:

- **Static Web Page**
- **Dynamic Web Page**

1) Static Web Page

A **Static Web Page** is one that displays **fixed content**. It remains the **same for every user**, regardless of who requests it.

◆ Key Features:

- **Content does not change** dynamically.
- **Same response** for all users.
- Created using **HTML/CSS only**.
- Stored directly on the **web server** and served as-is.

◆ File Extensions:

- .htm
- .html

◆ Memory Concept:

- Static refers to **continuous memory** – memory is **shared** across users.

2) Dynamic Web Page:

A **Dynamic Web Page** changes its content **based on user input**, database interactions, or other data.

◆ Key Features:

- Content is **generated dynamically** on request.
- Different users may get **different responses**.
- Built using **server-side scripting** (e.g., PHP, ASP.NET, JSP).

◆ File Extensions:

- .php, .asp, .aspx, .jsp

◆ Memory Concept:

- Dynamic refers to **discreet memory** – **new memory is allocated** for every user request.

Note: Every website that you designed starts with a default page called “index.html”.

Feature	Static Page	Dynamic Page
Content	Same for every user	Changes per request
Technology	HTML/CSS	PHP, ASP.NET, JSP, etc.
Speed	Very fast (pre-built)	Slightly slower (generated)
Extensions	.html, .htm	.php, .asp, .jsp, .aspx
Memory Usage	Continuous (shared)	Discreet (separate)
User Experience	Basic	Personalized & dynamic

HTML

HYPER TEXT MARKUP LANGUAGE

What is HTML?

- HTML stands for Hyper Text Markup Language.
- HTML is the standard markup language for creating Web pages.
- HTML describes the structure of a Web page.
- HTML elements tell the browser how to display the content.
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.
- Technically, HTML is a Markup language rather than a programming language.
- HTML is Not Case Sensitive.

❖ **Hyper Text:** Hyper Text means "Text within Text." A text has a link within it, is a hypertext.

❖ **Markup language:** A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic.

❖ **Developer:**

- HTML was developed by **Sir Tim Berners-Lee**, a British computer scientist, in 1990 while working at CERN (the European Organization for Nuclear Research). He is often credited as the inventor of the **World Wide Web**. Father of HTML is known as Sir Tim Berners-Lee.
- It is an open standard maintained by the World Wide Web Consortium (W3C).
- The latest version of HTML is HTML5.

❖ **HTML text editors:**

- Notepad(windows)
- Notepad++ (Windows)
- Visual Studio Code (Cross-platform)
- Sublime Text (Cross-platform)
- Atom (Cross-platform)
- Brackets (Cross-platform)
- Vim (Cross-platform)
- Emacs (Cross-platform)
- TextMate (Mac)
- Komodo Edit (Cross-platform)
- Bluefish (Cross-platform)
- Coda (Mac)
- UltraEdit (Windows, Mac, Linux)
- Dreamweaver (Windows, Mac)
- CoffeeCup HTML Editor (Windows, Mac)
- Pinegrow (Cross-platform)

Features of HTML (HyperText Markup Language)

1 Markup Language

- HTML is a **markup language** that uses **tags and elements** to define the **structure and presentation** of content on web pages.

2 Document Structure

- HTML organizes content using a **hierarchical structure** with elements like:

- `<html>`, `<head>`, `<body>`
- Semantic tags: `<header>`, `<footer>`, `<section>`

3 Hyperlinks

- With the `<a>` tag, HTML supports **hyperlinks** to:

- Navigate between internal pages
- Connect to external websites or files

4 Text Formatting

- Provides tags to style text, such as:

- Headings: `<h1>` to `<h6>`
- Paragraphs: `<p>`
- Bold: ``, ``
- Italics: `<i>`, ``

5 Lists

- HTML supports multiple list types:

- Ordered List: ``
- Unordered List: ``
- Definition List: `<dl>`

6 Images and Multimedia

- Supports embedding:

- **Images:** ``
- **Audio:** `<audio>`
- **Video:** `<video>`
- **External content:** `<iframe>`

7 Forms

- Used to collect **user input** via:

- `<form>`, `<input>`, `<select>`, `<textarea>`, `<button>`

8 Semantic Elements (HTML5)

- Provides **meaningful structure** using:

- `<header>`, `<nav>`, `<main>`, `<section>`, `<article>`, `<aside>`, `<footer>`

9 Responsive Design Support

- HTML supports **responsive layout techniques** (via meta viewport, CSS) to ensure websites work on **mobile, tablet, and desktop**.

10 Metadata

- HTML uses <meta> tags to provide information about the web page:
 - Character encoding (charset)
 - Description
 - Author
 - Viewport settings for mobile devices

1 1 Accessibility

- Enables support for **screen readers** and other assistive technologies using:
 - alt attribute for images
 - Semantic tags for better navigation

1 2 Cross-Browser Compatibility

- HTML allows pages to be displayed **consistently across all major web browsers** (Chrome, Firefox, Safari, Edge, etc.)

1 3 Embedded Scripting

- HTML can embed **JavaScript** using <script> to enable:
 - Form validation
 - Interactive features
 - Dynamic content rendering

1 4 Versioning

- HTML has evolved over time:
 - From HTML 1.0 to **HTML5 (latest)**
 - HTML5 includes new elements, multimedia support, APIs, and mobile responsiveness

Quick Remember & understand of HTML'S Features:

Feature	Description
Markup Language	Uses tags to define page structure
Hyperlinks	Navigate within/between websites
Text Formatting	Styles content using formatting tags
Lists	Creates ordered/unordered/definition lists
Multimedia Support	Embeds images, videos, audio, etc.
Forms	Collects user input
Semantic Elements (HTML5)	Adds meaning to structure for SEO & accessibility

Responsive Design	Supports mobile-friendly layouts
Metadata	Provides page info for browsers/search engines
Accessibility	Helps screen readers understand page structure
Cross-Browser Compatibility	Works across all browsers
Embedded Scripting	JavaScript for interactivity
Versioning	Latest version: HTML5

Structure of an HTML Page

An HTML document follows a **standard structure** that helps browsers understand how to display the content.

✓ HTML Page Has Two Main Sections:

1. Document Declaration
2. Document Scope

◆ 1. Document Declaration

The document declaration tells the browser which version of HTML is being used.

- It is placed at the **very top** of the page.
 - For HTML5, it is written as: `<!DOCTYPE html>`
 - If not declared, the browser assumes **HTML4**.
 - It helps browsers understand how to read and display the content.
 - It contains **meta information** like:
 - HTML **version**
 - **Language/Culture**
 - **License** or document type
- **Note:** `<!DOCTYPE html>` is **not an HTML tag** – it's a **declaration**.

◆ 2. Document Scope

The document scope is the actual HTML content of the page, enclosed in the `<html>...</html>` tags.

- It defines the **start and end of the HTML document**.
- It is essential for every HTML file.
- The **language of the page** is set using the lang attribute:

```
<html lang="en-in">

</html>
```

- **Note:** lang="en-in" tells the browser the page is in **English (India)**.

❖ Sections Inside Document Scope:

◆ A. <head>...</head> – Head Section

The <head> section contains **information *about* the page**, not content shown on the page.

It usually includes:

Tag	Purpose
<meta>	Provides metadata (like character set, viewport)
<title>	Sets the title shown on browser tab
<link>	Links to external CSS files
<style>	Internal CSS styles
<script>	JavaScript code for behavior/functionality

Important HTML Head Tags & Their Functions

◆ 1). <meta> Tag



Purpose:

- “Meta” means **metadata** (i.e., data about the page).
- Helps search engines and browsers understand the content of the page.
- Improves **SEO (Search Engine Optimization)**.
- Supports **responsive design** and **language encoding**.



Common Meta Tags Syntax:

```
<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">
```



Details:

➤ <meta charset="UTF-8">

- Defines the **character encoding** of the web page.
- **UTF-8** supports most languages, including English, Hindi, Japanese, Arabic, etc.

✓ About UTF-8:

- Full form: **Unicode Transformation Format – 8-bit**
- Developed by **Ken Thompson & Rob Pike** at Bell Labs in 1992.
- Supports characters in **1 byte to 4 bytes**.
 - English: 8-bit
 - Hindi, Chinese, Arabic: 16–32 bit
- Example: ॐ → **E0 A4 85**

✓ Character Encoding:

- Technique to **digitally represent** characters and symbols.
- Without it, browsers wouldn't know how to **display text** properly.

✓ ASCII (American Standard Code for Information Interchange) vs Unicode:

Feature	ASCII	Unicode (UTF-8)
Language	Only English	Almost all global languages
Size	7-bit or 8-bit	8-bit to 32-bit
Developer	ANSI	Unicode Consortium
Example Use	A = 65	ॐ = E0 A4 85

◆ 2). <title> Tag

✓ Purpose:

- Sets the **title of the web page** (shown in browser tab).
- Used when **bookmarking** the page.

✓ Syntax:

```
<title>My Website Title</title>
```

◆ 3). <link> Tag

✓ Purpose:

- Used to **link external files**, such as:
 - CSS stylesheets
 - Favicon icons

✓ Syntax:

```
<link rel="stylesheet" href="style.css">  
<link rel="icon" href="favicon.ico">
```


Attributes:

- rel: Specifies the relationship (e.g., "stylesheet", "icon")
- href: Specifies the **file path or URL**

◆ 4). <script> Tag

✅ Purpose:

- Used to include **JavaScript code** (either inside HTML or from external file).
- It is used to embed client or server-side script.
- Enables **interactivity** like form validation, animations, etc.

✅ Syntax:

```
<script src="script.js"></script>
```

◆ 5). <style> Tag

✅ Purpose:

- Used to write **internal CSS** directly in the HTML document.
- It is used to embed styles in web page.
- Controls **layout, colors, fonts**, etc.

✅ Syntax:

```
<style>

body {

    background-color: #f2f2f2;

    color: #333;

}
```

◆ B. <body>...</body> – Body Section

The <body> tag contains all the **content that will be visible on the web page** (text, images, links, etc.)

Contains the **visible content** of the page — what users see and interact with.

✅ Examples of content inside <body>:

- Headings (<h1> to <h6>)
- Paragraphs (<p>)
- Images ()
- Links (<a>)
- Buttons, Forms, Tables, etc.

❖ **Note:** Only the content in the <body> appears on the screen when you load the webpage.

Basic Structure:

```
<!DOCTYPE html>

<html>

  <head>

    <title>Page Title</title>

    <meta charset="UTF-8">

  </head>

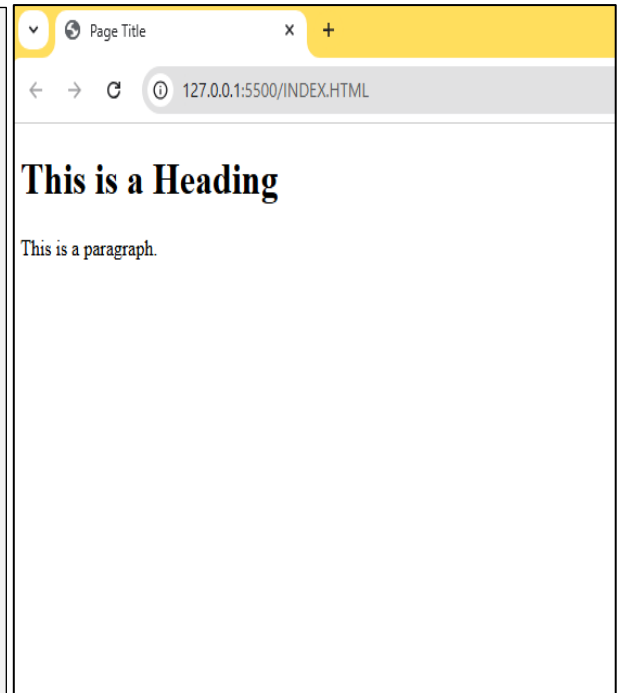
  <body>

    <h1>This is a Heading</h1>

    <p>This is a paragraph. </p>

  </body>

</html>
```



Explanation of HTML Structure Elements:

Tag	Name	Purpose
<!DOCTYPE html>	Document Type Declaration	Tells the browser this is an HTML5 document
<html>	Root Element	Encloses the entire HTML content
<head>	Head Section	Contains metadata, links to CSS/JS, and the title of the page
<title>	Title	Sets the title shown on the browser tab
<meta>	Metadata	Provides info like character set or viewport
<body>	Body Section	Contains all visible content on the web page like headings, paragraphs, images, etc.
<h1> to <h6>	Headings	Used for titles or subtitles (h1 = highest, h6 = lowest)
<p>	Paragraph	Used to write text content
<a>, , <div>, etc.	Other Tags	Used to add links, images, structure, etc.

HTML Page Structure

<!DOCTYPE html> ← Tells version of HTML

<html> ← HTML Root Element

<head> ← Used to contain page HTML metadata

<title>Page Title</title> ← Title of HTML page

</head>

<body> ← Hold content of HTML

<h2>Heading Content</h2> ← HTML heading tag

<p>Paragraph Content</p> ← HTML paragraph tag

</body>

</html>

Building Blocks of HTML

HTML is built on three core components:

1. Tags
2. Attributes
3. Elements

◆ 1. Tags

✓ Definition:

- **Tags** are the **basic building blocks** of HTML.
- They are enclosed within **angle brackets**: `< >`.
- Most tags come in **pairs**:
 - **Opening tag**: `<tagname>`
 - **Closing tag**: `</tagname>`

✓ Examples:

```
<h1>This is a Heading</h1>
<p>This is a paragraph.</p>
<div>This is a container</div>
```

◆ 2. Attributes

✓ Definition:

- **Attributes** provide **additional information** about an HTML element.
- Always specified in the **opening tag**.
- An attribute is made up of:
 - **Name**
 - **Value**
- Written as: `name="value"`

✓ Syntax:

```
<tagname attribute_name="attribute_value">Content</tagname>
```

✓ Examples:

```

<a href="https://example.com">Visit Site</a>
<input type="text" placeholder="Enter name">
```

Attribute Name	Purpose	Example
href	Specifies link URL	<code>Click</code>
src	Specifies image source	<code></code>
alt	Alternate text for image	<code></code>
type	Input type	<code><input type="password"></code>

◆ 3. Elements

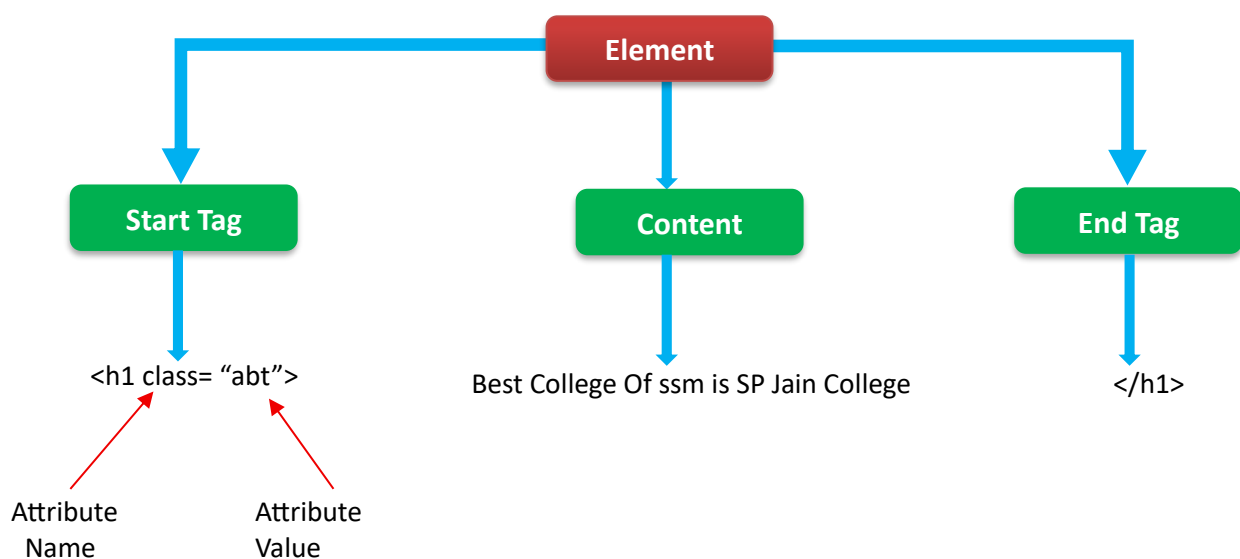
✅ Definition:

- An **HTML element** includes:
 - The **opening tag**
 - The **content**
 - The **closing tag**
- It is the **actual component** that defines content on the page.

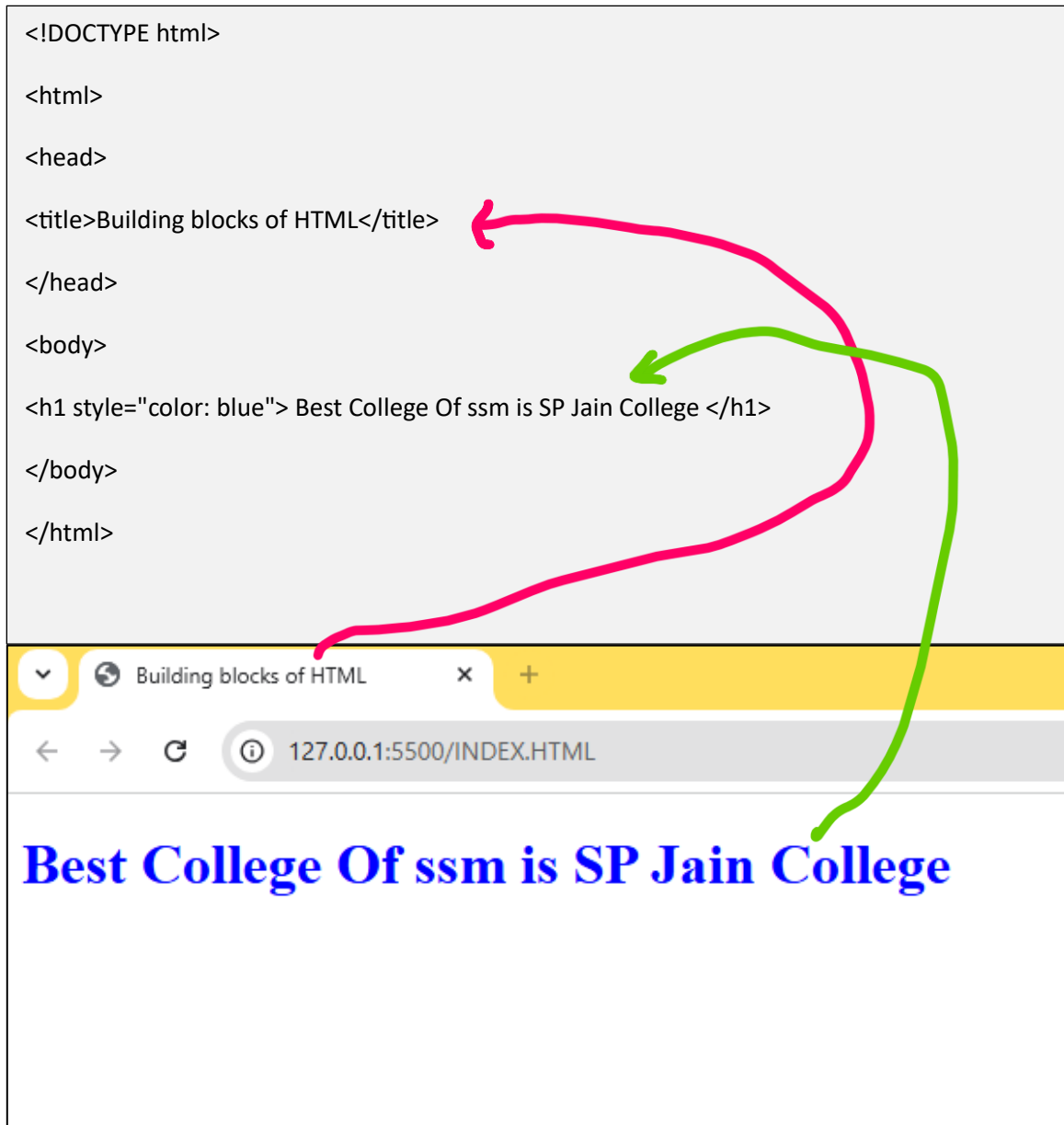
✅ Example:

```
<p>This is a paragraph element.</p>
```

- ◆ Everything inside `<p>` and `</p>` (including the tags themselves) is an **HTML element**.



Example:



To see the HTML code for any website, follow these methods:

Method 1: Using “View Page Source” (Quick Way)

1. **Right-click** on the webpage (anywhere).
2. Select **“View Page Source”** (or press Ctrl + U on Windows / Cmd + Option + U on Mac).
3. A new tab will open showing the **HTML source code** of the page.

Method 2: Using Developer Tools (Inspect)

1. **Right-click** on the specific element or area of the website.
2. Click on **“Inspect”** or **“Inspect Element”**.
3. The **Developer Tools panel** will open.
 - You'll see the HTML in the **Elements** tab.
 - You can also see **CSS, JavaScript**, and live edits.

HTML EDITORS

Editor Name	Type	Key Features	Best For
Visual Studio Code	Code Editor (Free)	Syntax highlighting, code completion, extensions, debugging tools	All levels of developers
Sublime Text	Code Editor (Paid with Free Trial)	Fast, lightweight, customizable, plugin support	Developers needing speed
Atom	Code Editor (Free)	Open-source, customizable UI, community plugins	Beginners and intermediate users
Notepad++	Text Editor (Free)	Lightweight, syntax highlighting, limited to Windows	Simple HTML editing on Windows
Brackets	Code Editor (Free)	Live preview, focused on web development, open-source	Web designers and front-end devs
Adobe Dreamweaver	IDE (Paid)	Visual design + code view, advanced tools, commercial product	Professional designers and teams
Emacs	Text Editor (Free)	Highly extensible, powerful with the right configuration	Advanced users and programmers
Vim	Text Editor (Free)	Terminal-based, efficient editing, customizable with plugins	Power users with experience
CodePen	Online Editor (Free/Paid)	Real-time preview, community sharing, good for HTML/CSS/JS snippets	Quick prototyping and inspiration
JSFiddle	Online Editor (Free)	Easy sharing, collaborative testing of HTML, CSS, JS	Testing small frontend code blocks
Repl.it (Replit)	Online IDE (Free/Paid)	Supports many languages, collaborative coding, cloud-based development	Learners, team projects, education

◆ 1. Visual Studio Code (VS Code)

- **Type:** Free, Open-source Code Editor
- **Platform:** Windows, macOS, Linux
- **Features:**
 - Smart code completion, IntelliSense
 - Syntax highlighting for HTML, CSS, JS, and more
 - Built-in Git support
 - Huge extension marketplace
 - Integrated terminal and debugging tools
- **Why Use:**
Ideal for professional and beginner developers due to rich features and active community.

◆ 2. Sublime Text

- **Type:** Paid (Free trial available)
- **Platform:** Windows, macOS, Linux
- **Features:**
 - Fast and responsive
 - Multiple selections and split editing
 - Command palette
 - Plugin support with Package Control
- **Why Use:**
Preferred by developers who need a lightweight and super-fast editor with customization options.

◆ 3. Atom

- **Type:** Free, Open-source
- **Platform:** Windows, macOS, Linux
- **Features:**
 - Created by GitHub
 - Built-in package manager
 - Smart autocompletes
 - Supports HTML, CSS, JS out of the box
- **Why Use:**
Great for those who want a customizable and user-friendly interface for web development.

◆ 4. Notepad++

- **Type:** Free, Open-source
- **Platform:** Windows only
- **Features:**
 - Lightweight and fast
 - Syntax highlighting for multiple languages
 - Tabbed document interface
 - Minimal system resource usage
- **Why Use:**
Simple tool for quick editing, great for beginners or making small edits to HTML code.

◆ 5. Brackets

- **Type:** Free, Open-source
- **Platform:** Windows, macOS, Linux
- **Features:**
 - Live preview in browser

- Preprocessor support (like LESS, SCSS)
- Focused on front-end development
- Inline editing
- **Why Use:**
Especially useful for web designers and frontend developers who want visual feedback while coding.

◆ 6. Adobe Dreamweaver

- **Type:** Paid (Commercial Software)
- **Platform:** Windows, macOS
- **Features:**
 - Visual (drag-and-drop) and code interface
 - Real-time browser preview
 - Git support
 - Integrated with Adobe ecosystem
- **Why Use:**
Ideal for designers and developers who prefer both design and code views, and work in teams or on larger projects.

◆ 7. Emacs

- **Type:** Free, Open-source
- **Platform:** Windows, macOS, Linux
- **Features:**
 - Extremely customizable with Emacs Lisp
 - Can be turned into a full IDE
 - Supports web dev tools with extensions
- **Why Use:**
Powerful tool for experienced developers who want complete control and extensibility.

◆ 8. Vim

- **Type:** Free, Open-source
- **Platform:** Windows, macOS, Linux
- **Features:**
 - Terminal-based editor
 - Keyboard-focused navigation
 - Customizable with .vimrc and plugins
 - Lightweight and fast
- **Why Use:**
For advanced users who value speed, efficiency, and work in terminal environments.

◆ 9. CodePen (Online Editor)

- **Type:** Free & Paid plans
- **Platform:** Web-based
- **Features:**
 - Live preview of HTML, CSS, JS
 - Showcase your work with the community
 - Embed pens into websites
 - Great UI for rapid prototyping
- **Why Use:**
Perfect for front-end developers to experiment, share, and showcase UI components or ideas.

◆ 10. JSFiddle (Online Editor)

- **Type:** Free
- **Platform:** Web-based
- **Features:**
 - Easy sharing and embedding
 - Supports multiple frameworks (jQuery, Vue, etc.)
 - Tabs for HTML, CSS, JS, and Output
- **Why Use:**
Best for testing small code snippets or demonstrating issues and solutions online.

◆ 11. Replit (Repl.it)

- **Type:** Free & Paid plans
- **Platform:** Web-based
- **Features:**
 - Supports multiple programming languages
 - Collaborative coding (pair programming)
 - Cloud-based, no setup needed
 - Useful for building full-stack apps
- **Why Use:**
Great for students and teams working on projects from any device with internet access.



How to Write HTML Code in Visual Studio Code

Step	Action	Explanation / Tip
1. Install VS Code	Go to https://code.visualstudio.com/ and download the installer for your OS.	Follow the installation instructions based on your operating system (Windows, macOS, or Linux).
2. Open VS Code	Launch the installed application.	You'll see a clean interface with a sidebar (Explorer, Extensions, etc.) and a large coding area.
3. Create a New File	Click File → New File or press Ctrl+N (Windows/Linux) or Cmd+N (macOS).	A new untitled editor tab will appear where you can begin writing code.
4. Save the File as HTML	Press Ctrl+S or Cmd+S and name your file index.html.	Use .html as the file extension so VS Code can enable HTML features like highlighting.
5. Write HTML Code	Type your HTML code in the editor.	Example: <pre><!DOCTYPE html> <html> <head> <title>My First Web Page</title> </head> <body> <h1>Hello, World!</h1> <p>This is a simple HTML page.</p> </body> </html></pre>
6. Use Auto-Completion	Start typing tags like <h1> and press Enter.	VS Code auto-closes tags and shows suggestions, saving time and reducing errors.
7. Format Code Properly	Right-click → Format Document or press Shift+Alt+F.	This neatly arranges your HTML code for readability.
8. Install & Use Live Server (Preview)	Open Extensions (Ctrl+Shift+X), search for Live Server , and install it. Then right-click your index.html file and click Open with Live Server .	This opens your HTML file in the browser and automatically refreshes when you save changes.
9. Save Regularly	Press Ctrl+S / Cmd+S often.	Prevent loss of work by saving your progress frequently.
10. Debug/Test HTML	Use browser Developer Tools (F12 in browser), or install extensions like HTMLHint for validation.	Helps you catch and fix errors in your code.
11. Enhance Your Page	Add more tags, CSS styles, or even JavaScript.	VS Code supports full web development (HTML + CSS + JS). Use extensions for emmet, snippets, or frameworks.

✓ How to Write HTML Code in Notepad (Step-by-Step)

Step	Action	Explanation / Tip
1. Open Notepad	<ul style="list-style-type: none">- Press Win + R, type notepad, press Enteror- Search Notepad from the Start menu	Notepad is a basic text editor that comes pre-installed with Windows.
2. Start a New Document	Click File → New	This opens a blank page where you can start writing code.
3. Write Your HTML Code	<p>Type your HTML content like this:</p> <pre>1 <!DOCTYPE html> 2 <html> 3 <head> 4 <title>My First Web Page</title> 5 </head> 6 <body> 7 <h1>Hello, Guys!</h1> 8 <p>This is a best College.</p> 9 </body> 10 </html> 11</pre>	This is the basic structure of an HTML document. You can type it directly into Notepad.
4. Save the HTML File	<ul style="list-style-type: none">- Press Ctrl + S or go to File → Save- Choose "All Files (.*)" in the "Save as type" option- Name the file index.html	The .html extension is important! If you save it as .txt, it won't work as a web page.
5. Open HTML File in a Browser	<ul style="list-style-type: none">- Go to the saved file location- Right-click → Open with → Choose browser (Chrome, Firefox, Edge, etc.)	Your web browser will open and display the web page based on the HTML code you wrote.
6. Edit and Update	<ul style="list-style-type: none">- Right-click the .html file → Open with → Notepad- Make changes and save (Ctrl + S) again- Refresh the browser to see updates	You can continue editing your webpage as needed and instantly view changes by refreshing the browser.

✓ How to Write HTML Code in Sublime Text (Step-by-Step)

Step	Action	Explanation / Tip
1. Install Sublime Text	Visit https://www.sublimetext.com/ and download the correct version for your OS.	Follow the installer instructions to complete the setup. It works on Windows, macOS, and Linux.
2. Open Sublime Text	Find and launch the Sublime Text editor from your applications list or desktop.	You'll see a clean, minimal code editor screen.
3. Create a New File	Click File → New File or press Ctrl+N (Windows/Linux) or Cmd+N (macOS).	This opens a new blank document for writing code.
4. Save the File as .html	Press Ctrl+S or Cmd+S → Choose your folder → Name the file index.html → Set file type to "All Files" if needed.	Saving with a .html extension is important to enable syntax highlighting and browser compatibility.
5. Write HTML Code	Type your HTML content in the editor. Example: <pre><!DOCTYPE html> <html> <head> <title>My First Web Page</title> </head> <body> <h1>Hello, World!</h1> <p>This is a simple HTML page.</p> </body> </html></pre>	Sublime will highlight syntax automatically once saved as .html.
6. Use Auto-Completion & Snippets	Start typing < or html and press Tab to activate Emmet abbreviation.	Emmet is built-in in Sublime for fast HTML coding. Try typing html:5 + Tab to get a complete HTML structure.
7. Save Your Changes	Press Ctrl+S / Cmd+S often to save changes.	Prevent data loss by saving frequently while editing.
8. Preview HTML File in Browser	Right-click the file from File Explorer/Finder → Open with your browser (Chrome, Firefox, etc.)	You'll see your HTML code rendered as a web page. You can refresh the browser after saving changes.
9. Continue Editing & Enhancing	Add more HTML tags, structure, links, images, etc.	Sublime supports many languages and has plugins to extend functionality for HTML, CSS, and JavaScript.

✓ HTML Tags: Basic Overview

- HTML tags are **keywords enclosed in angle brackets (<>)** that tell the browser how to display content.
- Most HTML tags come in pairs: an **opening tag <tag>** and a **closing tag </tag>**.
- Tags define structure (like paragraphs, headings, links) and behavior of elements.

📖 Syntax of HTML Tag

```
<tagname> Content goes here </tagname>
```

Example:

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

📖 Unclosed HTML Tags

These are **self-closing** (do not need a closing tag):

Tag	Meaning	Usage
 	Line Break	Adds a line break
<hr>	Horizontal Rule	Adds a horizontal line

Examples:

```
<p>This is line one.<br>This is line two.</p>  
<hr>
```

📖 Common Categories of HTML Tags

🔹 1. Meta Tags

These tags go inside the `<head>` section and define meta-information:

- `<!DOCTYPE>` – Defines the document type (HTML5)
- `<title>` – Sets the title of the page
- `<meta>` – Specifies metadata like character set or description
- `<style>` – Adds internal CSS
- `<link>` – Links external stylesheets

2. Text Formatting Tags

Tag	Purpose
<code><p></code>	Paragraph
<code><h1></code> to <code><h6></code>	Headings
<code></code>	Bold important text
<code></code>	Emphasized (italic) text
<code><abbr></code>	Abbreviation
<code><address></code>	Address info
<code><cite></code>	Citation
<code><code></code>	Code display
<code><ins></code> , <code></code>	Inserted or deleted text
<code><pre></code>	Preformatted text

3. Link Tags

- `<a>` – Defines a hyperlink
- `<base>` – Specifies base URL for all relative links

Example:

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

4. Image & Object Tags

- `` – Embed image
- `<map>`, `<area>` – Image mapping
- `<object>`, `<param>` – Multimedia

Example:

```

```

5. List Tags

- `` – Unordered list
- `` – Ordered list
- `` – List item
- `<dl>`, `<dt>`, `<dd>` – Description list

Example:

```
<ul>
  <li>HTML</li>
  <li>CSS</li>
</ul>
```

6. Table Tags

- `<table>` – Table container
- `<tr>` – Table row
- `<td>` – Table data
- `<th>` – Table header
- `<thead>`, `<tbody>`, `<tfoot>` – Groupings

Example:

```
<table>
  <tr><th>Name</th><th>Age</th></tr>
  <tr><td>John</td><td>25</td></tr>
</table>
```

7. Form Tags

- `<form>`, `<input>`, `<textarea>`, `<select>`, `<button>`, `<label>`, `<fieldset>`, `<legend>`

Example:

```
<form>
  Name: <input type="text" name="username">
</form>
```

8. Scripting Tags

- `<script>` – JavaScript
- `<noscript>` – Shown when JS is disabled

✓ HTML Attributes: Definition and Examples

- Provide **extra information** or **properties** about elements.
- Attributes are always specified in the **opening tag** of an element. (Written **inside the opening tag**.)
- Attributes are written as **name/value pairs**, like: `name="value"`

```
<tagname attribute="value">Content</tagname>
```

Attribute	Used In	Purpose	Example
href	<a>	URL of link	Visit
src		Image file path	
alt		Alternate text	
width, height	, <video>	Size control	
style	Any tag	Inline CSS styling	<p style="color: red;">Red Text</p>
lang	<html>	Language declaration	<html lang="en">
title	Most tags	Tooltip text	<p title="Info">Hover me</p>

Example:

```
1 <!DOCTYPE html>
2 <html lang="en"> <!-- 'Lang' attribute used here -->
3 <head>
4   <meta charset="UTF-8">
5   <title>HTML Attributes Example</title> <!-- 'title' element, not attribute -->
6 </head>
7 <body>
8
9   <h1 title="Main Heading">Welcome to My Page</h1> <!-- 'title' attribute -->
10
11  <p style="color: red;" title="This is a red paragraph.">This text is red using the style attribute.</p> <!-- 'style' and 'title' attributes -->
12
13  <a href="https://site.com" title="Go to Site">Visit Site</a> <!-- 'href' and 'title' attributes -->
14
15  <br><br>
16
17   <!-- 'src', 'alt', 'width', and 'height' -->
18
19 </body>
20 </html>
21
```

🔗 Explanation:

Attribute	Where Used	Effect
lang="en"	In <html>	Declares the language of the webpage
title="..."	On headings, links, and paragraphs	Shows a tooltip when hovered
style="color: red;"	On <p>	Changes paragraph text color
href="..."	On <a>	Makes the text a clickable link
src="..."	On 	Specifies image file path

alt="..."	On 	Shows alternate text if image fails
width, height	On 	Controls the image display size

Attribute	उपयोग कहाँ होता है	हिन्दी में मतलब	उदाहरण	समझाइए
href	<a> (anchor tag)	लिंक कहाँ जाना है	Visit	जब कोई user लिंक पर क्लिक करे, तो वो किस पेज पर जाए – ये href तय करता है।
src	, <video>, <audio>, <iframe>	स्रोत (Source)		ब्राउज़र को बताता है कि कौन सी फ़ाइल (जैसे इमेज या वीडियो) दिखानी है और कहाँ से लानी है।
alt		वैकल्पिक टेक्स्ट (Alternative text)		जब इमेज लोड नहीं होती, तब जो टेक्स्ट दिखाना है – उसे alt में लिखा जाता है। Screen reader भी इसे पढ़ते हैं।
width, height	, <video>	चौड़ाई और ऊँचाई		इमेज या वीडियो का आकार (size) तय करता है – कितनी चौड़ी और कितनी ऊँची दिखे।
style	किसी भी टैग में	CSS के ज़रिए स्टाइल देना	<p style="color: red;">Red text</p>	किसी भी HTML टैग का रंग, साइज, फॉन्ट वगैरह बदलने के लिए – इनलाइन स्टाइल देने के लिए।
lang	<html>	भाषा बताता है	<html lang="en">	यह बताता है कि वेबपेज किस भाषा में लिखा गया है – जैसे English (en), Hindi (hi) आदि।
title	किसी भी टैग में	Tooltip (सूचना जब hover करें)	<p title="NGO Info">This is Best College</p>	जब यूजर माउस से टेक्स्ट पर रुकता है, तो एक छोटा box (tooltip) खुलता है – वही title attribute का असर है।

◆ 1. Common Attributes (सामान्य एट्रिब्यूट्स)

Attribute	काम (Work)	उदाहरण (Example)	हिंदी में
id	Unique पहचान देता है	<div id="header">	एक यूनिक नाम देता है जिससे उसे CSS/JS में पहचाना जा सके
class	Styling या grouping के लिए	<p class="note">	एक जैसे elements को ग्रुप करता है
style	Inline CSS जोड़ता है	<h1 style="color:blue;">	सीधे HTML में ही स्टाइलिंग
title	Tooltip दिखाता है	<p title="Extra Info">Text</p>	माउस ले जाने पर जानकारी दिखती है

◆ 2. Form Attributes (फॉर्म से जुड़े एट्रिब्यूट्स)

Form Attributes:

- **name:** Identifies the name of an input field when submitting form data.
- **type:** Specifies the type of input element (e.g., text, password, checkbox).
- **value:** Sets the initial or default value for form elements.
- **placeholder:** Displays a short hint or example text in an input field.
- **required:** Requires that the user fill in the input field before submitting the form.
- **disabled:** Disables the input element, making it uneditable and unclickable.
- **readonly:** Makes the input element read-only (users can't edit it but can see its value).
- **form:** Associates an input element with a specific form by referencing the form's `id`.
- **maxlength:** Specifies the maximum number of characters allowed in a text field.
- **min` and `max:** Define the minimum and maximum values for numeric input fields.
- **pattern:** Sets a regular expression pattern for input validation.
- **autocomplete:** Controls whether browser autofill suggestions are enabled.
- **autofocus:** Automatically focuses the input element when the page loads.

Attribute	कार्य	उदाहरण	हिंदी में
name	डेटा भेजने के लिए नाम	<input name="email">	Form सबमिट करते वक्त सर्वर तक जाता है
type	Input का प्रकार	<input type="text">	टेक्स्ट, पासवर्ड, बटन आदि
value	Default वैल्यू सेट करता है	<input value="John">	पहले से भरी हुई जानकारी
placeholder	Hint दिखाता है	<input placeholder="Enter name">	अंदर हल्की ग्रे टेक्स्ट

required	Input भरना जरूरी	<input required>	भरे बिना form सबमिट नहीं होता
disabled	Element को बंद करता है	<input disabled>	यूजर कुछ टाइप नहीं कर सकता
readonly	केवल पढ़ने के लिए	<input readonly value="123">	यूजर देख सकता है पर बदल नहीं सकता
form	किसी specific form से जोड़ता है	<input form="myForm">	Input को किसी form से link करता है
maxlength	अधिकतम अक्षर	<input maxlength="10">	सिर्फ 10 characters टाइप कर सकते हैं
min, max	न्यूनतम और अधिकतम मान	<input type="number" min="1" max="10">	जैसे 1 से 10 तक ही मान्य
pattern	Regex pattern सेट करता है	<input pattern="[A-Za-z]{3,}">	जैसे कम से कम 3 अक्षर
autocomplete	Browser suggestions	<input autocomplete="on">	पहले भरी गई values दिखाता है
autofocus	पेज लोड पर फोकस	<input autofocus>	Automatically कर्सर इस input में आ जाता है

3. Link Attributes (<a> टैग)

Attribute	कार्य	उदाहरण
href	लिंक का URL	Google
target	लिंक कहां खुले	 – नई टैब में

4. Image Attributes (टैग)

Attribute	कार्य	उदाहरण
src	इमेज का स्रोत	
alt	वैकल्पिक टेक्स्ट	
width, height	इमेज साइज	

◆ 5. List Attributes (,)

Attribute	Element	उदाहरण
type		<ol type="A"> – A, B, C...
start		<ol start="5"> – 5, 6, 7... से शुरू

◆ 6. Table Attributes (<table>, <td>, <th>)

Attribute	कार्य	उदाहरण
border	बॉर्डर की मोटाई	<table border="1">
colspan	कॉलम मर्ज करता है	<td colspan="2">
rowspan	रो मर्ज करता है	<td rowspan="3">
scope	Header का स्कोप	<th scope="col">Name</th> – accessibility के लिए

◆ 7. Script Attributes (<script> टैग)

Attribute	कार्य	उदाहरण
src	JS फ़ाइल का URL	<script src="main.js">
type	स्क्रिप्ट का प्रकार	<script type="text/javascript">
defer	Script को HTML के बाद चलाना	<script defer src="main.js">
async	Script बिना रुकावट के चले	<script async src="main.js">

❖ Quick Summary of all attributes (हिंदी में):

Attribute	मतलब
href	लिंक कहां जाएगा
src	इमेज/वीडियो कहां से आएगा
alt	इमेज न दिखे तो क्या दिखे
style	कैसे दिखेगा
title	माउस ले जाने पर क्या दिखे
type, name, value	फॉर्म इनपुट को कंट्रोल करते हैं
required, read-only	भरना जरूरी या सिर्फ पढ़ने लायक
Col span, row span	टेबल cell मर्ज करने के लिए
autocomplete, autofocus	Input field के auto behaviors

✓ HTML Elements –

◆ 1. What is an HTML Element?

- An **HTML element** is the **building block** of an HTML document. It represents a piece of content (like text, images, links, etc.) and tells the browser **how that content should be displayed**.
- An HTML element consists of a **start tag**, **content**, and an **end tag**.
- An **HTML element** is everything from the **start tag to the end tag**, including the **content** in between.

Syntax:

```
<tagname>Content goes here...</tagname>
```

Example:

```
<p>This is a paragraph.</p>  
<h1>Welcome to My Website</h1>
```

- `<p>` is the start tag
 - This is a paragraph. is the content
 - `</p>` is the end tag
- Together, they form a **complete HTML element**.

◆ Types of HTML Elements:

1. Normal Elements (with opening & closing tag):

```
<h1>Hello</h1>  
<p>Paragraph</p>
```

2. Empty Elements (no closing tag):

- **Empty elements** in HTML are those elements that **do not contain any content** and **do not require a closing tag**.
- They are **self-contained** and are also called **void elements**.

```
<br> <!-- Line break -->  
<hr> <!-- Horizontal line -->  
 <!-- Image tag -->
```

Examples:-

Tag	Purpose	Example	Hindi Meaning
<code>
</code>	Inserts a line break	Hello World	लाइन ब्रेक डालता है
<code><hr></code>	Inserts a horizontal line	<hr>	क्षैतिज रेखा बनाता है
<code></code>	Displays an image		इमेज दिखाने के लिए
<code><input></code>	Creates input field	<input type="text">	इनपुट बॉक्स बनाता है
<code><meta></code>	Defines meta info	<meta charset="UTF-8">	पेज की जानकारी देता है
<code><link></code>	Links CSS file	<link rel="stylesheet" href="style.css">	CSS फाइल जोड़ता है
<code><source></code>	Defines media source	<source src="video.mp4">	ऑडियो/वीडियो का स्रोत

3. Nested Elements (elements inside elements):

- **Nesting** in HTML means **placing one HTML element inside another**.
- It helps structure content and allows you to **combine formatting and meaning** together.

 **Syntax:**

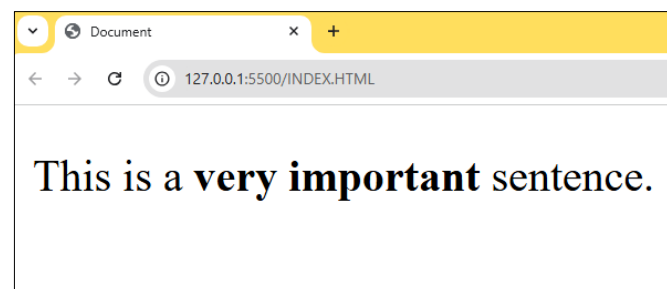
```
<outer-element>
  <inner-element>Content</inner-element>
</outer-element>
```

◆ **Example:**

```
<p>This is a <strong>very important</strong> sentence.</p>
```

<p> is the outer element
 is the inner element nested inside it

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   <p>This is a <strong>very important</strong> sentence.</p>
10
11 </body>
12 </html>
13
```



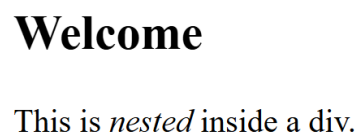
◆ **Real-Life Example: -**

```
<div>
  <h2>Welcome</h2>
  <p>This is <em>nested</em> inside a div.</p>
</div>
```

div contains both h2 and p
em is also nested inside p

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   <div>
10    <h2>Welcome</h2>
11    <p>This is <em>nested</em> inside a div.</p>
12  </div>
13
14 </body>
15 </html>
16
```

output →



✓ Block-Level & Inline HTML Elements

◆ Block-Level Elements

- Block-level elements are those that **start on a new line** and **take up the full width** available (from left to right of the page).

Key Features:

- Always starts on a **new line**
- Takes **full width** of the container
- Can contain **both block-level and inline elements**
- Used for **layout structure**

Examples:

`<div>`, `<p>`, `<h1>` to `<h6>`, ``, ``, ``, `<table>`, `<section>`, `<article>`, `<nav>`, `<footer>`, `<form>`

`<address>`, `<article>`, `<aside>`, `<blockquote>`, `<canvas>`, `<dd>`, `<div>`, `<dl>`, `<dt>`, `<fieldset>`, `<figcaption>`, `<figure>`, `<footer>`, `<form>`, `<h1>`-`<h6>`, `<header>`, `<hr>`, ``, `<main>`, `<nav>`, `<noscript>`, ``, `<output>`, `<p>`, `<pre>`, `<section>`, `<table>`, `<tfoot>`, `` and `<video>`.

Example in Code:

```
<p>This is a paragraph.</p>
<h1>This is a heading</h1>
<div>
  <p>Paragraph inside div</p>
</div>
```

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   <p>This is a paragraph.</p>
10  <h1>This is a heading</h1>
11  <div>
12    <p>Paragraph inside div</p>
13  </div>
14
15 </body>
16 </html>
17
```

This is a paragraph.

This is a heading

Paragraph inside div

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   <h3 style="background-color: blue;"> this is h3 heading</h3>
10  <div style="background-color: green;"> this is div</div>
11  <p style="background-color: red;"> this is paragraph</p>
12 </body>
13 </html>
14
```

this is h3 heading

this is div

this is paragraph

◆ Inline Elements

- Inline elements are those that do not start on a new line and only take up as much width as necessary.

Key Features:

- Flows **within the line**
- Takes **only as much space** as needed
- Cannot contain block-level elements
- Used for **formatting content**

Examples:

🚩 Inline Elements:

ये एलिमेंट उसी लाइन में रहते हैं और केवल उतनी जगह लेते हैं जितनी उन्हें जरूरत होती है।

◆ विशेषताएं:

- नई लाइन से शुरू नहीं होता
- उतनी ही चौड़ाई लेता है जितनी जरूरत हो
- ब्लॉक एलिमेंट नहीं रख सकता

◆ उदाहरण:

``, `<a>`, ``, ``, ``

``, `<a>`, ``, ``, ``, `<input>`, `<label>`, `<abbr>`, `
`

🌐 Example in Code:

```
<p>This is <strong>bold</strong> and <em>italic</em> text.</p>
<a href="#">Click Here</a>
```

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9 <p>This is <strong>bold</strong> and <em>italic</em> text.</p>
10 <a href="#">Click Here</a>
11
12 </body>
13 </html>
14
```

Document

127.0.0.1:5500/INDEX.HTML

This is **bold** and *italic* text.

[Click Here](#)

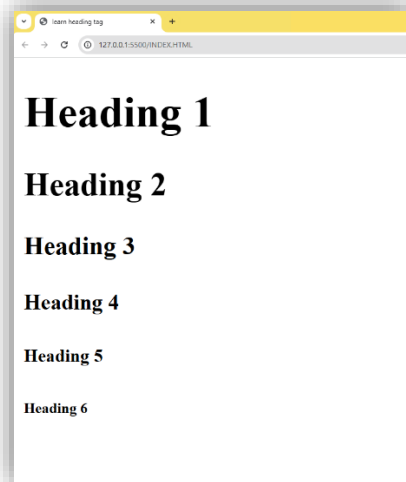
HTML HEADING TAGS

In HTML, **heading tags** are used to define headings/Title/Subtitle on a webpage. They range from <h1> to <h6>, where:

- <h1> is the **highest (or most important)** level heading.
- <h6> is the **lowest (or least important)** level heading.
- These tags help organize content and improve readability. Search engines also use them to understand the structure of your page content for SEO (Search Engine Optimization).

Example:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>learn heading tag</title>
6 </head>
7 <body>
8 <h1>Heading 1</h1>
9 <h2>Heading 2</h2>
10 <h3>Heading 3</h3>
11 <h4>Heading 4</h4>
12 <h5>Heading 5</h5>
13 <h6>Heading 6</h6>
14 </body>
15 </html>
```

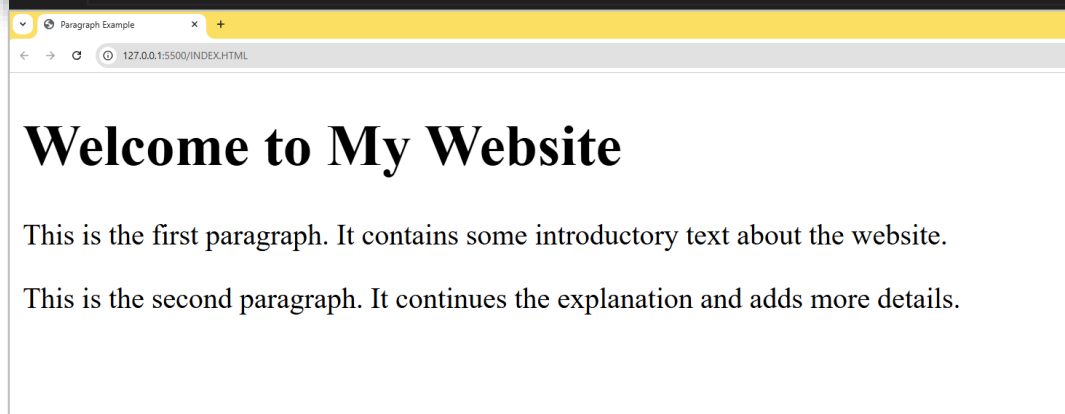


HTML PARAGRAPH TAGS

- HTML paragraphs are defined with the <p> tag.
- Browsers automatically add space before and after a paragraph.

Example:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Paragraph Example</title>
5 </head>
6 <body>
7
8 <h1>Welcome to My Website</h1>
9 <p>This is the first paragraph. It contains some introductory text about the website.</p>
10
11 <p>This is the second paragraph. It continues the explanation and adds more details.</p>
12
13 </body>
14 </html>
15
```



SPACE INSIDE HTML PARAGRAPH TAGS

By default, **HTML collapses multiple spaces and line breaks** inside a `<p>` tag. That means even if you put many spaces or press Enter (new line) in your HTML code, the browser shows it as a **single space**.

Example (won't work as expected):

```
<p>This      is      spaced      text.</p>
```

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Paragraph Example</title>
5 </head>
6 <body>
7   <p>This      is      spaced      text.</p>
8 </body>
9 </html>
10
```

Paragraph Example
127.0.0.1:5500/INDEX.HTML

This is spaced text.

✓ **To preserve spaces and line breaks, use:**

1. ` ` for multiple spaces

Example:

```
<p>This&nbsp;&nbsp;&nbsp;has&nbsp;&nbsp;&nbsp;extra&nbsp;&nbsp;&nbsp;spaces.</p>
```

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Paragraph Example</title>
5 </head>
6 <body>
7   <p>This&nbsp;&nbsp;&nbsp;has&nbsp;&nbsp;&nbsp;extra&nbsp;&nbsp;&nbsp;spaces.</p>
8 </body>
9 </html>
10
```

Paragraph Example
127.0.0.1:5500/INDEX.HTML

This has extra spaces.

2. `
` for line breaks

Example:

```
<p>This is line one.<br>This is line two.</p>
```

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Paragraph Example</title>
5 </head>
6 <body>
7   <p>This is line one.<br>This is line two.</p>
8 </body>
9 </html>
10
```

Paragraph Example
127.0.0.1:5500/INDEX.HTML

This is line one.
This is line two.

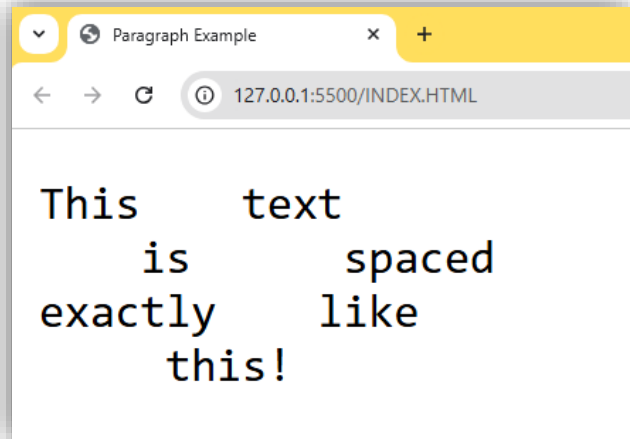
✓ HTML `<pre>` Element

- The `<pre>` tag is used to display **preformatted text**. It:
- Preserves spaces, tabs, and line breaks.
- Uses a **monospaced** (fixed-width) font.

Syntax:

```
<pre>
This   text
      is      spaced
exactly like
      this!
</pre>
```

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 |   <title>Paragraph Example</title>
5 </head>
6 <body>
7 <pre>
8 This   text
9 |     is      spaced
10 exactly like
11 |     this!
12 </pre>
13 </body>
14 </html>
15
```



◆ HTML TEXT FORMATTING

HTML provides tags to format text:

Tag	Description	Example
<code></code>	Bold text	<code>Bold</code>
<code></code>	Important text	<code>Strong</code>
<code><i></code>	Italic text	<code><i>Italic</i></code>
<code></code>	Emphasized text	<code>Emphasis</code>
<code><mark></code>	Highlighted text	<code><mark>Marked</mark></code>
<code><u></code>	Underlined text	<code><u>Underline</u></code>
<code><small></code>	Smaller text	<code><small>Small text</small></code>
<code></code>	Deleted text	<code>Deleted</code>
<code><ins></code>	Inserted text	<code><ins>Inserted</ins></code>
<code><sub></code>	Subscript text	H <code><sub>2</sub></code> O
<code><sup></code>	Superscript text	X <code><sup>2</sup></code>

Example: -

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>HTML Text Formatting Tags</title>
```

```
</head>
```

```
<body>
```

```
<p><b>This is bold text using &lt;b>&lt;/b></p>
```

```
<p><strong>This is important text using &lt;strong>&lt;/strong></p>
```

```
<p><i>This is italic text using &lt;i>&lt;/i></p>
```

```
<p><em>This is emphasized text using &lt;em>&lt;/em></p>
```

```
<p><mark>This is highlighted text using &lt;mark>&lt;/mark></p>
```

```
<p><u>This is underlined text using &lt;u>&lt;/u></p>
```

```
<p><small>This is smaller text using &lt;small>&lt;/small></p>
```

```
<p><del>This is deleted text using &lt;del>&lt;/del></p>
```

```
<p><ins>This is inserted text using &lt;ins>&lt;/ins></p>
```

```
<p>This is water formula: H<sub>2</sub>O (using &lt;sub>&lt;/sub>)</p>
```

```
<p>This is X<sup>2</sup> (using &lt;sup>&lt;/sup>)</p>
```

```
</body>
```

```
</html>
```

Paragraph Example

127.0.0.1:5500/INDEX.HTML

This is bold text using

This is important text using

This is italic text using <i>

This is emphasized text using

This is highlighted text using <mark>

This is underlined text using <u>

This is smaller text using <small>

This is deleted text using

This is inserted text using <ins>

This is water formula: H₂O (using <sub>)

This is X² (using <sup>)

```
7 <!DOCTYPE html>
```

```
8 <html>
```

```
9 <head>
```

```
10 <title>HTML Text Formatting Tags</title>
```

```
11 </head>
```

```
12 <body>
```

```
13
```

```
14 <p><b>This is bold text using &lt;b>&lt;/b></p>
```

```
15
```

```
16 <p><strong>This is important text using &lt;strong>&lt;/strong></p>
```

```
17
```

```
18 <p><i>This is italic text using &lt;i>&lt;/i></p>
```

```
19
```

```
20 <p><em>This is emphasized text using &lt;em>&lt;/em></p>
```

```
21
```

```
22 <p><mark>This is highlighted text using &lt;mark>&lt;/mark></p>
```

```
23
```

```
24 <p><u>This is underlined text using &lt;u>&lt;/u></p>
```

```
25
```

```
26 <p><small>This is smaller text using &lt;small>&lt;/small></p>
```

```
27
```

```
28 <p><del>This is deleted text using &lt;del>&lt;/del></p>
```

```
29
```

```
30 <p><ins>This is inserted text using &lt;ins>&lt;/ins></p>
```

```
31
```

```
32 <p>This is water formula: H<sub>2</sub>O (using &lt;sub>&lt;/sub>)</p>
```

```
33
```

```
34 <p>This is X<sup>2</sup> (using &lt;sup>&lt;/sup>)</p>
```

```
35
```

```
36 </body>
```

```
37 </html>
```

```
38
```

```
39 </body>
```

```
40 </html>
```

HTML QUOTATION AND CITATION ELEMENTS

Tag	Purpose	Example
<blockquote>	Defines a block quote	<blockquote>This is a block quote.</blockquote>
<q>	Short inline quote	<q>Short quote</q>
<abbr>	Abbreviation	<abbr title="HyperText Markup Language">HTML</abbr>
<cite>	Cites a work (book, film, etc.)	<cite>The Jungle Book</cite>
<address>	Author/contact info	<address>Written by John</address>

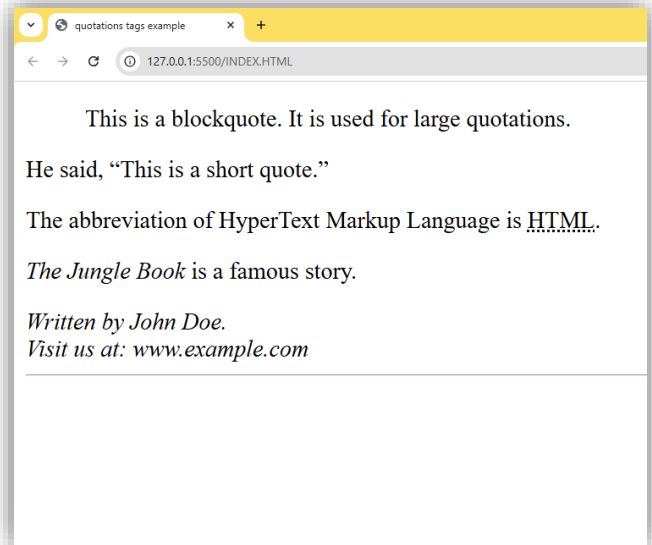
```
<blockquote>
  This is a blockquote. It is used for large quotations.
</blockquote>

<p>He said, <q>This is a short quote.</q></p>

<p>The abbreviation of HyperText Markup Language is
  <abbr title="HyperText Markup Language">HTML</abbr>.
</p>

<p><cite>The Jungle Book</cite> is a famous story.</p>

<address>
  Written by John Doe. <br>
  Visit us at: www.example.com
</address>
```



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>quotations tags example</title>
5 </head>
6 <body>
7   <blockquote>
8     This is a blockquote. It is used for large quotations.
9   </blockquote>
10
11   <p>He said, <q>This is a short quote.</q></p>
12
13   <p>The abbreviation of HyperText Markup Language is
14     <abbr title="HyperText Markup Language">HTML</abbr>.
15   </p>
16
17   <p><cite>The Jungle Book</cite> is a famous story.</p>
18
19   <address>
20     Written by John Doe. <br>
21     Visit us at: www.example.com
22   </address>
23
24   <hr>
25 </body>
26 </html>
27
```



HTML PHRASE TAG

- Phrase tags are semantic text-level elements.
- The HTML phrase tags are special purpose tags, which defines the structural meaning of a block of text or semantics of text.

- Abbreviation tag : **<abbr>**
- Definition tag: **<dfn>**
- Quoting tag: **<blockquote>**
- Short quote tag : **<q>**
- Code tag: **<code>**
- Keyboard tag: **<kbd>**
- Address tag: **<address>**
- Renders in italic: **<cite>**
- Quoted from another source: **<blockquote>**

Tag	Description
	Emphasized text (italics)
	Important text (bold)
<abbr>	Abbreviation
<dfn>	Definition term
<code>	Programming code
<samp>	Sample output
<kbd>	Keyboard input
<var>	Variable name

Example:

```
<p>This is an <em>emphasized</em> phrase.</p>
<p>This is a <strong>strong</strong> phrase.</p>
<p><abbr title="World Health Organization">WHO</abbr> was founded in 1948.</p>
<p><dfn>HTML</dfn> is the standard markup language for web pages.</p>
<p>This is computer <code>code</code> inside a paragraph.</p>
<p>Output is <samp>Hello World</samp></p>
<p>Press <kbd>Ctrl</kbd> + <kbd>S</kbd> to save.</p>
<p>The formula is: <var>x</var> + <var>y</var></p>
```

This is an *emphasized* phrase.

This is a **strong** phrase.

WHO was founded in 1948.

HTML is the standard markup language for web pages.

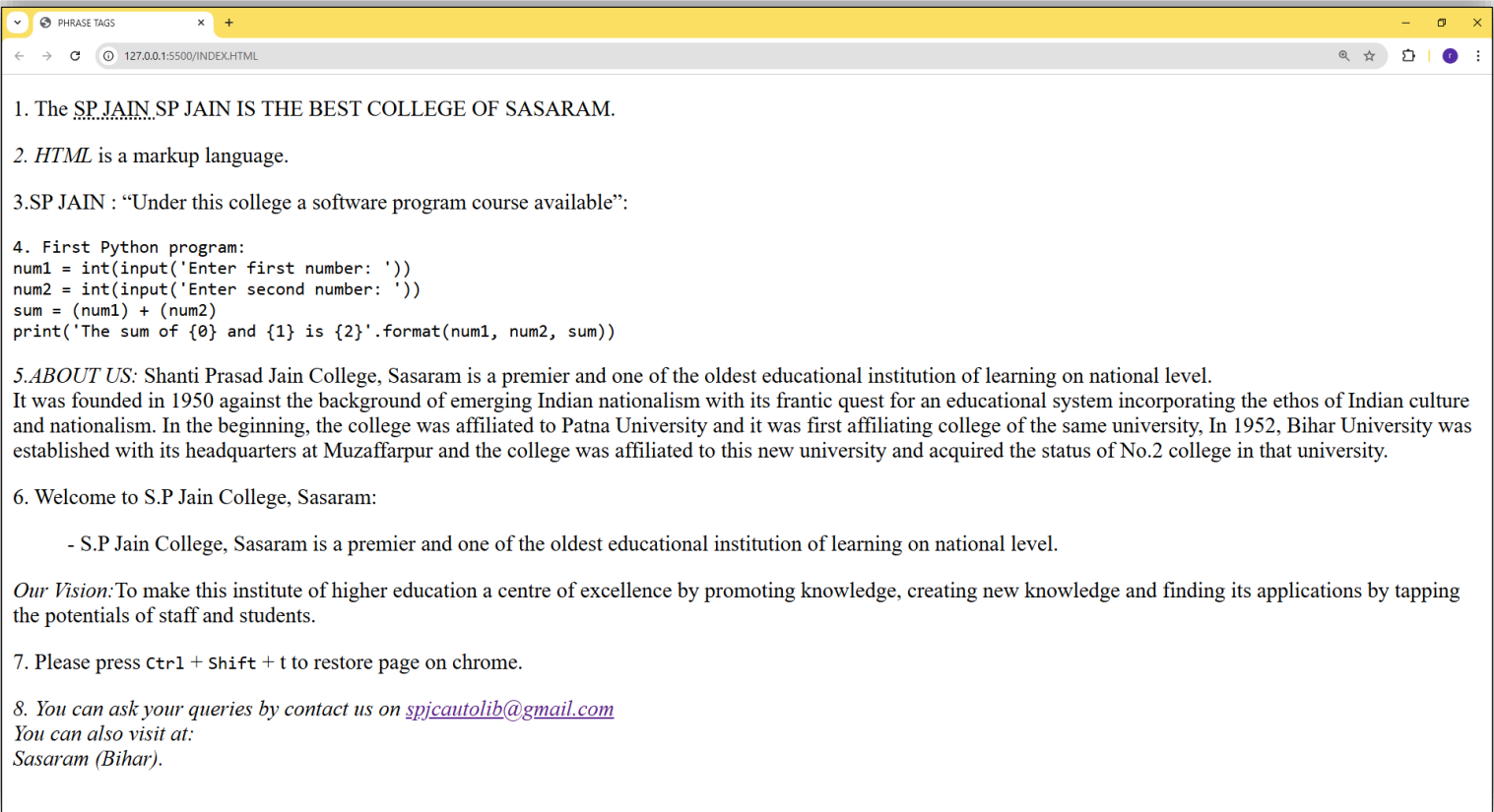
This is computer code inside a paragraph.

Output is Hello World

Press Ctrl + S to save.

The formula is: x + y

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>PHRASE TAGS</title>
6 </head><body>
7 <p>1. The <abbr title = "Research Innovation and Development">SP JAIN </abbr> SP JAIN IS THE BEST COLLEGE OF SASARAM.
8 <p><dfn>2. HTML </dfn> is a markup language. </p>
9 <p>3.SP JAIN : <q>Under this college a software program course available</q></p>
10 <pre>4. First Python program: <code>
11 num1 = int(input('Enter first number: '))
12 num2 = int(input('Enter second number: '))
13 sum = (num1) + (num2)
14 print('The sum of {0} and {1} is {2}'.format(num1, num2, sum)) </code>
15 </pre>
16 <p><cite>5.ABOUT US: </cite> Shanti Prasad Jain College, Sasaram is a premier and one of the oldest educational institution of learning on national
17 level.
18 <br>
19 It was founded in 1950 against the background of emerging Indian nationalism with its frantic quest for an educational system incorporating the
20 ethos of Indian culture and nationalism. In the beginning, the college was affiliated to Patna University and it was first affiliating college of
21 the same university, In 1952, Bihar University was established with its headquarters at Muzaffarpur and the college was affiliated to this new
22 university and acquired the status of No.2 college in that university.</p>
23 <p>6. Welcome to S.P Jain College, Sasaram:</p>
24 <blockquote cite="https://spjc.autolib.org/">
25 - S.P Jain College, Sasaram is a premier and one of the oldest educational institution of learning on national level.
26 </blockquote>
27 <p><cite>Our Vision:</cite>To make this institute of higher education a centre of excellence by promoting knowledge, creating new knowledge and
28 finding its applications by tapping the potentials of staff and students. </p>
29 <p>7. Please press <kbd>Ctrl</kbd> + <kbd>Shift</kbd> + t<kbd></kbd> to restore page on chrome.</p>
30 <address>8. You can ask your queries by contact us on <a href="">spjcautolib@gmail.com</a>
31 <br> You can also visit at: <br> Sasaram (Bihar).
32 </address>
33 </body></html>
```



HTML COMMENT TAG

Used to insert comments in the HTML code. Comments are not displayed in the browser.

Syntax:

```
<!-- This is a comment -->
```

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>comment tag</title>
6 </head>
7 <body>
8 <!-- This is a comment. It will not be displayed in the browser -->
9 <p>This is a paragraph with a hidden comment above.</p>
10 </body>
11 </html>
```



HTML STYLES

Used to apply CSS styles directly to HTML elements using the style attribute.

Syntax:

```
<tag name style="property: value;">
```

```
<p style="color: blue; font-size: 18px;">This is styled text.</p>
```

Where: The property is a **CSS property**. The value is a **CSS value**.

1. **Background Color:**

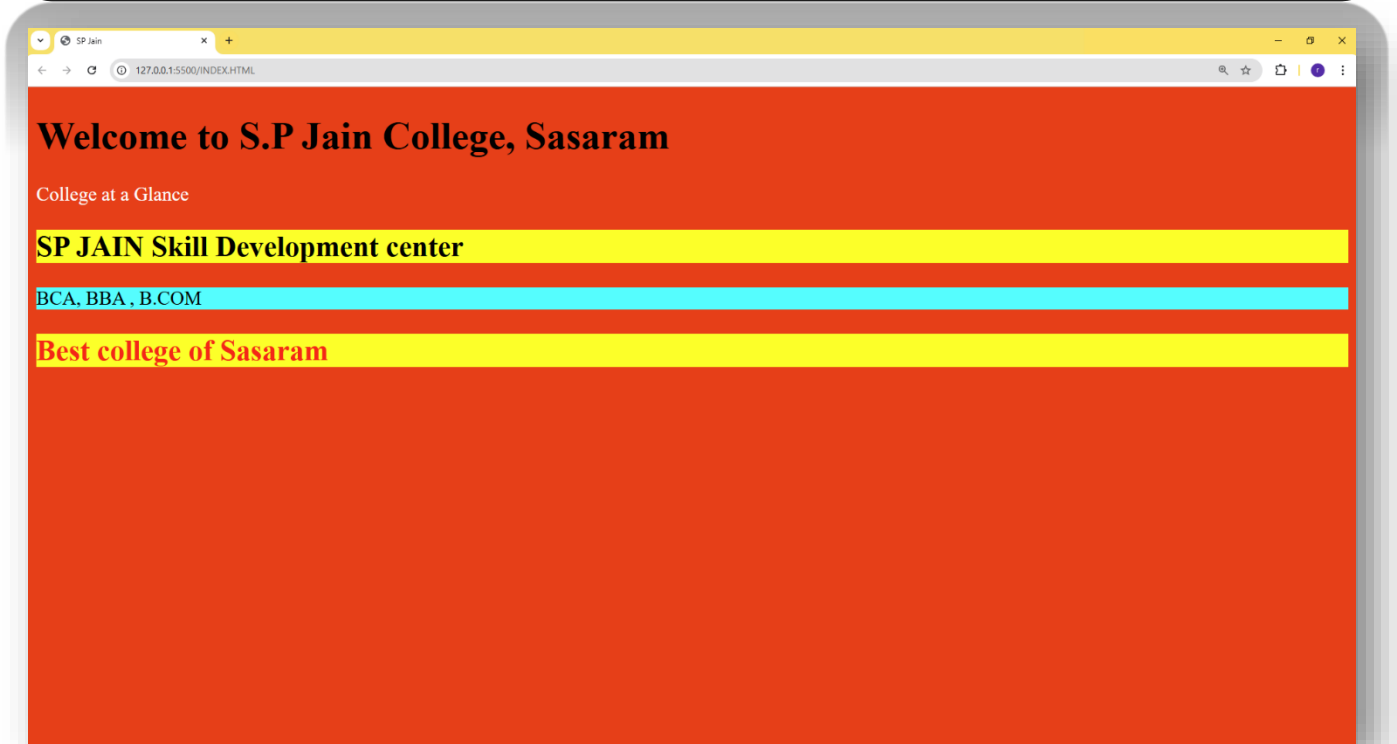
The CSS background-color property defines the background color for an HTML element.

2. **Text Color:**

The CSS color property defines the text color for an HTML element:

Example: -

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>SP Jain</title>
6 </head>
7 <body style="background-color: #c00000;">
8 <h1>Welcome to S.P Jain College, Sasaram </h1>
9 <P style="color: white;">College at a Glance</P>
10 <h2 style="background-color: yellow;">SP JAIN Skill Development center</h2>
11 <P style="background-color: cyan;">BCA, BBA , B.COM </P>
12 <h2 style="background-color: yellow; color: red;">Best college of Sasaram</h2>
13 </body>
14 </html>
```



3. **font-family:** for text fonts
4. **font-size:** for text sizes
5. **text-align:** for text alignment

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>SP JAIN </title>
6 </head>
7 <body >
8 <h1 style="font-family: verdana;">This is a heading</h1>
9 <p style="font-family: courier;">This is a paragraph. </p>
10 <h1 style="font-size:250%;">This is a heading</h1>
11 <p style="font-size:180%;">This is a paragraph.</p>
12 <h1 style="text-align: center;">Centered Heading</h1>
13 <p style="text-align: center;">Centered paragraph. </p>
14 </body>
15 </html>
```

