# Government Girls Polytechnic Bilaspur

**Subject Name:** Static Web Programming
**Subject Code:** 2033372(033)
**Semester :** 3rd

Prepared By:
Mr. Nuresh Kumar Dewangan
Lecturer
Department of Computer Science & Engineering

**Web Design Principles and Basics**

## **Basic Principles Involved in Building a Website** (वेबसाइट बनाने में शामिल मूलभूत सिद्धांत)

### 1. Planning (योजना बनाना)

Website बनाने का पहला और सबसे महत्वपूर्ण step है **planning**. इस stage में decide किया जाता है कि website का **purpose** क्या है — information देना, products बेचना, या services provide करना।

- Audience कौन है (students, customers, general public)।
- कौन-कौन से pages होंगे (Home, About, Contact, Gallery, etc.)।
- Website का structure और navigation कैसा रहेगा।

### 2. Design (डिज़ाइनिंग)

Design phase में focus किया जाता है **user interface (UI)** और **user experience (UX)** पर। Website visually appealing होनी चाहिए और साथ ही user-friendly भी। Main principles of web design:

- **Simplicity (सरलता)** – Design साफ़ और uncluttered हो।
- **Consistency (सुसंगतता)** – Font, color, layout हर page पर similar हो।
- **Responsiveness** – Website mobile, tablet, और desktop पर ठीक से दिखनी चाहिए।
- **Readability** – Text clearly visible और readable हो।

### 3. Content Creation (सामग्री निर्माण)

Website की soul उसका content होता है।

- Text, images, videos और infographics सभी relevant और updated होने चाहिए।
- Content को SEO (Search Engine Optimization) friendly बनाना ज़रूरी है ताकि Google जैसे search engines पर website आसानी से मिले।
- Heading tags (H1, H2, H3), meta descriptions और keywords का सही use करें।

### 4. Development (विकास प्रक्रिया)

इस stage में actual website बनती है using web technologies जैसे:

- **HTML** – Structure बनाने के लिए
- **CSS** – Design और styling के लिए
- **JavaScript** – Dynamic behavior और interactivity के लिए

Frameworks और libraries (जैसे Bootstrap, React, Django) का use development को तेज़ और organized बनाता है।

**5. Testing (परीक्षण)**

Website को launch करने से पहले thoroughly test किया जाता है। Testing types:

- **Functional Testing** – सभी links और buttons ठीक से काम कर रहे हैं या नहीं।
- **Compatibility Testing** – Website हर browser और device पर सही चल रही है या नहीं।
- **Performance Testing** – Load time और speed check करना।
- **Security Testing** – Website सुरक्षित है या नहीं, hacking से बचाव के लिए।

**6. Deployment (प्रकाशन)**

Testing complete होने के बाद website को **web server** या **hosting platform** पर upload किया जाता है ताकि users इसे internet पर access कर सकें। Domain name (जैसे www.example.com) assign किया जाता है और DNS settings configure की जाती हैं।

**7. Maintenance (रखरखाव)**

Website launch के बाद भी regular maintenance ज़रूरी है।

- Content और software updates करना।
- Broken links fix करना।
- Security patches apply करना।
- User feedback के आधार पर improvements करना।

## **Planning Process in Website Development** (वेबसाइट विकास में योजना बनाने की प्रक्रिया)

**Introduction (परिचय)**

Website development का सबसे पहला और सबसे महत्वपूर्ण step है — **Planning (योजना बनाना)**।
अगर planning clear और structured हो, तो पूरा development process smooth रहता है और website अपने लक्ष्य को सफलतापूर्वक पूरा करती है।
Planning phase में हम यह decide करते हैं कि website **क्यों**, **किसके लिए**, और **कैसे** बनाई जाएगी।

**1. Define the Purpose and Goals (उद्देश्य और लक्ष्य तय करना)**

सबसे पहले यह स्पष्ट करना ज़रूरी है कि website का **purpose** क्या है। कुछ सामान्य उद्देश्यों के उदाहरण:

- Educational purpose (जैसे college/institute websites)
- Business purpose (product selling, service promotion)
- Informational purpose (blogs, news sites)

इसके बाद, **specific goals** set किए जाते हैं जैसे:

- Visitors बढ़ाना
- Online sales बढ़ाना
- Feedback collect करना
- Information sharing improve करना

**2. Identify the Target Audience (लक्षित दर्शक पहचानना)**

Website design और content audience के अनुसार तय किया जाता है। उदाहरण:

- अगर audience students हैं, तो website में study material, timetable और notices होने चाहिए।
- अगर business clients हैं, तो products, pricing और testimonials दिखाने ज़रूरी हैं।

Target audience की पहचान के लिए इन factors पर ध्यान दिया जाता है:

- Age group
- Education level
- Technical knowledge
- Language preference
- Device usage (mobile/desktop)

**3. Research and Competitor Analysis (अनुसंधान और प्रतियोगी विश्लेषण)**

Planning phase में research करना बहुत महत्वपूर्ण है।

- Similar websites को analyze करें कि वे क्या अच्छा कर रही हैं और क्या कमी है।
- Market trends और user expectations को समझें।
- इससे आपकी website में **unique features** और **better usability** लाने में मदद मिलती है।

**4. Define the Website Structure (वेबसाइट की संरचना तय करना)**

इस step में website का **blueprint** तैयार किया जाता है जिसे **Site Map** कहा जाता है। Site map में यह तय होता है कि website में कौन-कौन से pages होंगे और उनका आपस में क्या relation होगा।
उदाहरण:

- Home
- About Us
- Services
- Gallery
- Contact Us

**Benefits of a site map:**

- Developer को structure समझने में आसानी
- Navigation clear रहती है
- Future expansion आसान होता है

**5. Decide the Content Requirements (सामग्री की योजना बनाना)**

हर page के लिए कौन-सा content चाहिए — यह planning stage में तय किया जाता है। Content types:

- Text (information, description)
- Images और graphics
- Videos या animations
- Documents या downloadable files

**SEO (Search Engine Optimization)** के लिए keywords भी इसी चरण में plan किए जाते हैं ताकि future में website easily searchable हो।

**6. Select Technology and Platform (तकनीक और प्लेटफॉर्म का चयन)**

Planning के दौरान यह भी तय किया जाता है कि website किस technology पर बनाई जाएगी:

- **Static Website** → HTML, CSS, JavaScript
- **Dynamic Website** → PHP, Python (Django/Flask), Node.js, etc.
- **CMS (Content Management System)** → WordPress, Joomla, Drupal

इसके साथ ही domain name और hosting service का भी चयन किया जाता है।

**7. Design Layout and Navigation Plan (डिज़ाइन लेआउट और नेविगेशन योजना)**

Website के look and feel की rough sketch या **wireframe** बनाई जाती है। इससे यह तय होता है कि:

- Menus कहाँ होंगे
- Buttons और forms कैसे दिखेंगे
- Content placement और color theme कैसी होगी

Navigation user-friendly होना चाहिए ताकि visitor आसानी से हर page तक पहुँच सके।

**8. Budget and Time Estimation (बजट और समय का अनुमान)**

Planning के दौरान यह भी तय करना ज़रूरी है कि:

- Website बनाने में कितना खर्च आएगा (design, hosting, domain, maintenance आदि)
- Development में कितना समय लगेगा
- कौन-कौन सी resources (team members, tools) आवश्यक होंगे

Proper budgeting से project delay और extra cost से बचा जा सकता है।

**9. Risk Analysis and Backup Plan (जोखिम विश्लेषण और वैकल्पिक योजना)**

हर project में कुछ risks होते हैं जैसे:

- Time delay
- Budget overrun
- Technical issues
- Data loss

इसलिए पहले से **backup plans** तैयार रखना चाहिए, जैसे:

- Regular backups
- Alternate hosting
- Testing schedules

**10. Approval and Documentation (स्वीकृति और दस्तावेज़ीकरण)**

Final step में पूरी planning को document किया जाता है और stakeholders से approval लिया जाता है।
इस documentation में शामिल होता है:

- Project scope
- Timeline
- Resource allocation
- Technical specifications

यह document आगे development team के लिए **guide** का काम करता है।

## **Five Golden Rules of Web Designing** (वेब डिज़ाइनिंग के पाँच स्वर्णिम नियम)

### **Introduction (परिचय)**

Web designing केवल सुंदर layout बनाने तक सीमित नहीं है, बल्कि इसका मुख्य उद्देश्य है — **user को एक बेहतर experience देना**।
एक अच्छी website वही होती है जो देखने में आकर्षक हो, इस्तेमाल में आसान हो, और अपने उद्देश्य को पूरा करे।
इसी के लिए कुछ मूलभूत सिद्धांतों को "**Five Golden Rules of Web Designing**" कहा जाता है।

### **1. Keep It Simple and Consistent (सरलता और सुसंगतता बनाए रखें)**

**Explanation:**
User को website पर जानकारी जल्दी और आसानी से मिलनी चाहिए। अगर design बहुत complex या confusing होगा, तो user site छोड़ देगा।
Simple design का मतलब है –

- Clutter-free layout
- Limited colors and fonts
- Clean typography

**Consistency** का अर्थ है कि हर page पर design elements (fonts, color schemes, navigation menus) एक जैसे दिखें।

**Example:**
अगर "Contact Us" बटन हर page पर एक ही जगह और एक ही color में है, तो user को उसे ढूंढने में दिक्कत नहीं होगी।

**Key Points:**

- हर page पर समान layout रखें।
- White space (खाली जगह) का सही उपयोग करें।
- Unnecessary animations या sound effects से बचें।

### **2. Easy Navigation (सुव्यवस्थित नेविगेशन)**

**Explanation:**
Navigation किसी भी website की **backbone** होती है।
User को किसी भी page पर जाने के लिए clear और simple path मिलना चाहिए।

**Good navigation** का मतलब है कि user 2–3 clicks में अपनी information तक पहुँच जाए।

**Key Principles:**

- Main menu हर page पर दिखाई दे।
- Important links (Home, About, Contact) easily accessible हों।
- Breadcrumbs या navigation bar का use करें ताकि user को पता चले कि वह कहाँ है।

**Example:**
E-commerce website में "Home → Category → Product → Checkout" जैसी navigation path user को आसानी देता है।

### 3. Mobile Friendly and Responsive Design (मोबाइल फ्रेंडली और उत्तरदायी डिज़ाइन)

**Explanation:**
आज ज्यादातर users websites को mobile या tablet पर access करते हैं। इसलिए responsive design ज़रूरी है ताकि website **हर screen size पर perfectly adjust हो सके**।

**Techniques for responsiveness:**

- Use of CSS frameworks जैसे **Bootstrap**
- Flexible images और fluid grids
- Avoid fixed-width layouts

**Benefits:**

- Better user experience
- Higher search engine ranking (Google mobile-friendliness को SEO में महत्व देता है)

**Example:**
अगर किसी site पर desktop में 3-column layout है, तो mobile view में वही design 1-column में automatically adjust होना चाहिए।

### 4. Fast Loading and Performance Optimization (तेज़ लोडिंग और प्रदर्शन सुधार)

**Explanation:**
अगर website load होने में 3 सेकंड से ज़्यादा लगाती है, तो user उसे छोड़ सकता है। इसलिए website की speed को optimize करना जरूरी है।

**Ways to improve speed:**

- Images को compress करें।
- Unused CSS/JS files को हटाएँ।
- Caching और Content Delivery Network (CDN) का इस्तेमाल करें।
- Lightweight design frameworks का use करें।

**Benefits:**

- Better user satisfaction
- Improved SEO ranking
- Reduced bounce rate

**Example:**
Google PageSpeed Insights या GTmetrix जैसे tools से performance regularly check करें।

**5. Readable and Accessible Content (पढ़ने योग्य और सुलभ सामग्री)**

**Explanation:**
Website का सबसे महत्वपूर्ण भाग उसका **content** होता है। अगर text पढ़ने में कठिन है या background के साथ contrast नहीं बना रहा, तो user site पर ज़्यादा देर नहीं रहेगा।

**Key Principles:**

- Font size readable रखें (16px या अधिक)।
- Background और text color में proper contrast रखें।
- Headings (H1, H2, H3) और bullet points का उपयोग करें।
- Alt text, captions और ARIA labels का उपयोग करके site को visually impaired users के लिए accessible बनाएं।

**Example:**
Dark background पर light text या vice versa readability को बेहतर बनाता है।

## **Designing Navigation Bar** (नेविगेशन बार का डिज़ाइन)

**Introduction (परिचय)**

Navigation bar किसी भी website का **सबसे महत्वपूर्ण हिस्सा** होता है। यह user को website के अलग-अलग pages पर जाने का रास्ता दिखाता है। एक अच्छा navigation bar user experience (UX) को बेहतर बनाता है और website को professional look देता है।

Navigation bar को design करते समय ध्यान रखना चाहिए कि user **minimum clicks में desired page** तक पहुँच सके।

**1. Definition (परिभाषा)**

**Navigation Bar** या **Menu Bar** एक user interface element होता है जो website के शीर्ष (header) या किनारे (sidebar) में स्थित होता है।
इसमें site के मुख्य sections के links होते हैं जैसे:

- Home
- About Us
- Services
- Gallery
- Contact Us

**2. Importance of Navigation Bar (नेविगेशन बार का महत्व)**

- यह user को site structure समझने में मदद करता है।
- Information तक पहुँचने का आसान तरीका प्रदान करता है।
- SEO (Search Engine Optimization) को सुधारता है क्योंकि proper navigation से search engines pages को आसानी से crawl कर सकते हैं।
- User retention बढ़ाता है — अगर navigation आसान होगा, तो user website पर ज़्यादा समय बिताएगा।

**3. Types of Navigation Bars (नेविगेशन बार के प्रकार)**

**(a) Horizontal Navigation Bar (क्षैतिज नेविगेशन बार)**

- सबसे common प्रकार है।
- Usually page के top पर होता है।
- Tabs या buttons एक पंक्ति में arranged रहते हैं।
- Suitable for websites with limited main sections.

**Example:**

Home | About | Services | Gallery | Contact

**(b) Vertical Navigation Bar (ऊर्ध्वाधर नेविगेशन बार)**

- Sidebar के रूप में left या right side में रहता है।
- Suitable for content-heavy websites जैसे dashboards या admin panels।
- Scrollable design के साथ sub-menu भी शामिल हो सकता है।

**(c) Drop-down Navigation (ड्रॉप-डाउन नेविगेशन)**

- जब user किसी menu item पर hover करता है या click करता है, तो sub-menu items दिखाई देते हैं।
- Useful for categorizing related pages.

**Example:**
"Services" → Web Design, SEO, Hosting

### (d) Sticky or Fixed Navigation (स्थायी नेविगेशन)

- Scroll करने पर भी screen के top पर fixed रहता है।
- User हमेशा navigation options देख सकता है।
- Modern and user-friendly design.

### (e) Hamburger Menu (हैमबर्गर मेनू)

- Mobile view में commonly used।
- तीन horizontal lines के icon पर click करने से menu खुलता है।
- Space-saving design for small screens.

## 4. Design Principles for Navigation Bar (डिज़ाइन सिद्धांत)

*(a) Clarity and Simplicity (स्पष्टता और सरलता)*

- Menu items के नाम short और meaningful होने चाहिए।
- Avoid complex terms — "About Us", "Contact", "Products" जैसे familiar words इस्तेमाल करें।

*(b) Consistency (सुसंगतता)*

- Navigation हर page पर same position में हो।
- Color scheme, font style, and hover effects consistent रहें।

*(c) Visual Hierarchy (दृश्य अनुक्रम)*

- Important links पहले दिखाएँ।
- Dropdown menus में parent-child relationship clear रखें।
- Use highlighting या bold text to indicate the current page.

*(d) Responsiveness (उत्तरदायी डिज़ाइन)*

- Navigation हर device (desktop, tablet, mobile) पर सही दिखना चाहिए।
- Mobile screens के लिए hamburger menu या collapsible navbar का उपयोग करें।

*(e) Accessibility (सुलभता)*

- Keyboard navigation support करें (Tab key से navigate किया जा सके)।
- Links में **alt text** और **ARIA labels** जोड़ें ताकि visually impaired users के लिए accessible हो।

### 5. Common Mistakes to Avoid (सामान्य गलतियाँ जिनसे बचना चाहिए)

- बहुत ज़्यादा menu items डालना।
- Navigation को hidden या confusing बनाना।
- Hover effects बहुत flashy बनाना।
- Inconsistent spacing और font size रखना।

### 6. Best Practices (सर्वोत्तम उपाय)

- Navigation हमेशा **top या left side** में रखें।
- Current page को highlight करें (active state)।
- Menu items logically grouped रखें।
- Search box को navigation में शामिल करें।
- Test navigation on multiple devices and browsers.

## **Page Design in Web Development** (वेब विकास में पेज डिज़ाइन)

### Introduction (परिचय)

Page Design website development का एक अत्यंत महत्वपूर्ण चरण है।
यह निर्धारित करता है कि website का प्रत्येक page **कैसा दिखेगा**, **कैसे कार्य करेगा**, और **user को कैसा अनुभव देगा**।
एक अच्छा page design न केवल website को आकर्षक बनाता है बल्कि **usability** और **user satisfaction** को भी बढ़ाता है।

### 1. Meaning of Page Design (पेज डिज़ाइन का अर्थ)

Page Design का अर्थ है — किसी webpage की **layout**, **color scheme**, **text arrangement**, **images**, और **interactive elements** को इस प्रकार व्यवस्थित करना कि पूरा page visually appealing और functionally effective हो।

इसमें HTML structure, CSS styling और visual components का logical combination शामिल होता है।

### 2. Objectives of Page Design (पेज डिज़ाइन के उद्देश्य)

- Website को **सौंदर्यपूर्ण (visually attractive)** बनाना
- Information को **सुव्यवस्थित (well-organized)** ढंग से प्रस्तुत करना
- User को **सुगमता (ease of use)** प्रदान करना
- Different devices (desktop, mobile, tablet) पर **responsive layout** देना
- Branding और theme consistency बनाए रखना

## 3. Key Elements of Page Design (पेज डिज़ाइन के प्रमुख घटक)

*(a) Layout (लेआउट)*

Layout तय करता है कि content कहाँ और किस प्रकार से दिखेगा।
एक balanced layout user को naturally content पढ़ने और explore करने में मदद करता है।

**Types of Layouts:**

- **Fixed layout:** Constant width, mostly for desktops
- **Fluid layout:** Width adjusts with screen size
- **Grid layout:** Multiple sections arranged in columns and rows
- **Single-page layout:** All information on one scrollable page

*(b) Color Scheme (रंग संयोजन)*

Color website की identity और mood तय करते हैं।

- Background और text में contrast होना चाहिए ताकि readability बढ़े।
- Primary, secondary और accent colors carefully choose करें।
- Too many bright colors avoid करें।

**Example:**
Educational websites – light blue, white (calm and professional look)
E-commerce websites – red, orange (attention-grabbing colors)

*(c) Typography (टाइपोग्राफी)*

Font selection readability और appearance दोनों को प्रभावित करता है।

- Font style simple और clean रखें (जैसे Roboto, Open Sans)।
- Headings और paragraph fonts में difference रखें।
- Proper line spacing और font size (16px or more) का ध्यान रखें।

*(d) Images and Graphics (चित्र और ग्राफिक्स)*

Visual elements attention attract करते हैं और message को जल्दी convey करते हैं।

- High-quality images इस्तेमाल करें।
- Size optimize करें ताकि page slow न हो।
- Decorative images से ज़्यादा informative visuals पर ध्यान दें।

*(e) White Space (खाली स्थान)*

White space या negative space वो जगह होती है जो content के बीच में खाली छोड़ी जाती है। यह readability बढ़ाती है और design को clutter-free रखती है।

*(f) Navigation (नेविगेशन)*

Page पर navigation links स्पष्ट और accessible होने चाहिए।
User को हर समय पता रहना चाहिए कि वह site के किस भाग में है और आगे कहाँ जा सकता है।

**4. Principles of Good Page Design (अच्छे पेज डिज़ाइन के सिद्धांत)**

1. **Balance (संतुलन):**
   Elements को symmetry या proportion में रखें ताकि design स्थिर लगे।
2. **Alignment (संरेखण):**
   Text और visuals एक logical grid पर aligned हों ताकि professional look मिले।
3. **Contrast (विपरीतता):**
   Colors, size, और shapes में अंतर design को attractive बनाता है।
4. **Consistency (संगति):**
   Font, color, button style हर page पर एक जैसा होना चाहिए।
5. **Emphasis (मुख्य बिंदु):**
   Important information (जैसे heading या call-to-action button) prominent दिखनी चाहिए।

**5. Responsive Page Design (उत्तरदायी पेज डिज़ाइन)**

आज के समय में users विभिन्न devices पर website access करते हैं — mobile, tablet, laptop, desktop।
Responsive design ensures कि हर screen size पर layout automatically adjust हो।

**Techniques:**

- CSS media queries का उपयोग करें।
- Flexible grids और images अपनाएँ।
- Bootstrap जैसी frameworks से responsive design आसान बनता है।

**6. Page Load Optimization (पेज लोड समय सुधारना)**

- Images को compress करें।
- External scripts और plugins को minimize करें।
- Browser caching enable करें।
- Lightweight design frameworks use करें।

Fast loading pages better user experience और high SEO ranking देते हैं।

### 7. Accessibility and Usability (सुलभता और उपयोगिता)

- Text और background में proper contrast रखें।
- Alt text का उपयोग करें ताकि visually impaired users को सहायता मिले।
- Keyboard navigation support दें।
- Buttons और links clearly visible और clickable हों।

### 8. Testing and Feedback (परीक्षण और प्रतिक्रिया)

Design तैयार होने के बाद pages को test करना आवश्यक है —

- Different browsers (Chrome, Firefox, Edge) में check करें।
- Different screen sizes पर view करें।
- Users से feedback लें और आवश्यक सुधार करें।

## **Home Page Layout in Web Design** (वेब डिज़ाइन में होम पेज लेआउट)

### Introduction (परिचय)

Website का **Home Page** किसी भी site का **पहला और सबसे महत्वपूर्ण page** होता है।
यह site का **मुख्य द्वार (main entrance)** होता है, जहाँ से user को पूरी website की दिशा मिलती है।
एक well-designed home page न केवल website की पहचान बनाता है, बल्कि **first impression** भी तय करता है।

Home Page Layout इस बात पर निर्भर करता है कि information को किस प्रकार से व्यवस्थित और प्रस्तुत किया गया है ताकि visitor को तुरंत site का उद्देश्य समझ आ जाए।

### 1. Meaning of Home Page Layout (होम पेज लेआउट का अर्थ)

**Home Page Layout** का अर्थ है — वेबसाइट के मुख्य पृष्ठ की **रचना (structure)** और **डिज़ाइन व्यवस्था (arrangement)**।
इसमें तय किया जाता है कि कौन-से sections कहाँ होंगे, कौन-से रंग और font उपयोग होंगे, और navigation कैसा रहेगा।

यह layout पूरे website के design और theme का **base structure** होता है।

### 2. Importance of Home Page (होम पेज का महत्व)

- Website का **पहला impression** बनाता है।
- User को बताता है कि site में क्या-क्या है।

- Easy navigation प्रदान करता है।
- Branding और organization की पहचान दिखाता है।
- Visitors को आगे explore करने के लिए motivate करता है।

### 3. Objectives of Home Page Layout (होम पेज लेआउट के उद्देश्य)

- Information को logical और attractive तरीके से प्रस्तुत करना।
- Clear navigation प्रदान करना।
- Call-to-action (CTA) जैसे "Learn More", "Register Now", "Contact Us" को highlight करना।
- Visitors को quickly समझाना कि site का उद्देश्य क्या है।
- Responsive और fast-loading design देना।

### 4. Basic Structure of Home Page (होम पेज की मूल संरचना)

एक typical home page में निम्नलिखित मुख्य भाग होते हैं:

*(a) Header Section (हेडर भाग)*

- Page के सबसे ऊपर होता है।
- इसमें **logo**, **navigation bar**, और कभी-कभी **search bar** या **login/signup buttons** होते हैं।
- Header हर page पर consistent रहना चाहिए।

**Example:**
Logo (left side) → Navigation menu (center/right side) → Login button (top right corner)

*(b) Hero Section (मुख्य आकर्षण भाग)*

- यह सबसे ऊपर दिखने वाला **visual highlight area** होता है।
- इसमें high-quality image, background video या slider हो सकता है।
- अक्सर इसमें एक **main headline**, short description और **call-to-action button** होता है।

**Example:**
"Empowering Education through Technology"
→ Button: "Explore Courses"

*(c) About or Introduction Section (परिचय भाग)*

- इस भाग में organization, company या website के बारे में संक्षेप में बताया जाता है।
- Text के साथ image या icon add करके presentation आकर्षक बनाते हैं।
- Link "Read More" या "About Us" page के लिए दिया जाता है।

*(d) Services or Features Section (सेवाएँ / विशेषताएँ)*

- यह section बताता है कि website या organization क्या-क्या प्रदान करती है।
- Usually cards या icons के रूप में 3–4 main features दिखाए जाते हैं।

**Example:**
📄 Web Design | 📱 App Development | ☁️ Cloud Hosting | 🎓 Training

*(e) Testimonials or Reviews (प्रशंसा/समीक्षा भाग)*

- Users या clients की feedback दिखाने के लिए उपयोग किया जाता है।
- यह विश्वास (trust) बनाने में मदद करता है।

*(f) Latest News / Gallery / Highlights Section (ताज़ा जानकारी या गतिविधियाँ)*

- Recent updates, photos या announcements दिखाने के लिए।
- Educational websites में "Notices", "Events", या "Achievements" यहाँ दिखाए जाते हैं।

*(g) Contact or Footer Section (संपर्क / फ़ुटर भाग)*

Footer page का अंतिम भाग होता है, जिसमें शामिल होते हैं:

- Contact details
- Address
- Social media links
- Copyright information
- Quick links (Home, Privacy Policy, Terms & Conditions)

Footer simple और consistent होना चाहिए ताकि हर page से contact आसानी से किया जा सके।

**5. Design Principles for Home Page Layout (डिज़ाइन सिद्धांत)**

1. **Clarity (स्पष्टता):**
   User को तुरंत समझ में आना चाहिए कि site किस बारे में है।
2. **Visual Hierarchy (दृश्य अनुक्रम):**
   Important content (headline, CTA) सबसे पहले दिखाई दे।
3. **Consistency (संगति):**
   Font, color, और button styles हर page पर same रहें।

4. **Responsive Design (उत्तरदायी डिज़ाइन):**
   Home page हर device पर properly adjust हो।
5. **Speed Optimization (गति अनुकूलन):**
   Heavy graphics और unnecessary animations से बचें।
6. **Whitespace (खाली जगह का प्रयोग):**
   Design को साफ़ और uncluttered बनाए रखें।

### 6. Technical Aspects (तकनीकी पहलू)

- HTML का उपयोग structure के लिए।
- CSS का उपयोग layout और styling के लिए।
- JavaScript या frameworks (जैसे Bootstrap) का उपयोग interactive effects के लिए।
- Responsive behavior के लिए media queries का उपयोग करें।

### 7. Common Mistakes to Avoid (बचने योग्य गलतियाँ)

- बहुत ज़्यादा text या cluttered design।
- Auto-playing videos या popups जो user को distract करें।
- Confusing navigation या hidden menu।
- Inconsistent color theme।
- CTA (Call to Action) buttons को छिपाना या dull बनाना।

### 8. Example of a Simple Home Page Layout (सरल होम पेज लेआउट का उदाहरण)

```
-------------------------------------------------
| LOGO    | HOME | ABOUT | SERVICES | CONTACT US |
-------------------------------------------------
|  [HERO IMAGE / SLIDER]                        |
|  "Welcome to Our Institute"                   |
|  [Explore More]                               |
-------------------------------------------------
| ABOUT US SECTION (Brief intro + image)        |
-------------------------------------------------
| SERVICES SECTION (Icons or Cards)             |
-------------------------------------------------
| TESTIMONIALS SECTION (User feedback)          |
-------------------------------------------------
| CONTACT / FOOTER                              |
| Address | Phone | Email | Social Links        |
-------------------------------------------------
```

### 9. Example Technologies for Implementation

- **HTML5** for structure

- **CSS3 / Bootstrap** for layout and responsiveness
- **JavaScript / jQuery** for dynamic elements
- **Font Awesome** for icons
- **AOS (Animate on Scroll)** for smooth animations

## Design Concepts (डिज़ाइन की अवधारणाएँ)

### Introduction (परिचय)

Website designing में "Design Concepts" का मतलब है — ऐसे सिद्धांत और नियम जो एक वेबसाइट को **attractive**, **functional**, और **user-friendly** बनाते हैं।
Good design न केवल देखने में सुंदर होता है बल्कि user को website navigate करने में आसान बनाता है।
Design concepts का मुख्य उद्देश्य है — **visual harmony (दृश्य संतुलन)**, **clarity (स्पष्टता)** और **usability (उपयोगिता)** बनाए रखना।

### 1. Balance (संतुलन)

Balance का अर्थ है webpage के सभी elements — जैसे text, images, buttons, videos आदि का सही proportion में distribution करना।
दो प्रकार के balance होते हैं:

- **Symmetrical Balance (समान संतुलन)** – Elements दोनों ओर बराबर रूप से व्यवस्थित रहते हैं, जैसे mirror image layout।
- **Asymmetrical Balance (असमान संतुलन)** – Elements बराबर नहीं होते, पर visual weight ऐसा रखा जाता है कि layout फिर भी balanced लगे।
  उदाहरण: एक side पर image और दूसरी side पर text block।

### 2. Contrast (विपरीतता)

Contrast का उपयोग important elements को highlight करने के लिए किया जाता है।
Contrast बनाने के तरीके:

- Colors (Dark vs Light)
- Size (Large heading vs Small text)
- Font weight (Bold vs Regular)
- Shapes या background differences
  Contrast से design में **readability** और **focus** दोनों बढ़ता है।

### 3. Alignment (संरेखण)

Alignment सुनिश्चित करता है कि webpage पर हर element एक proper visual connection में रहे।
Proper alignment से layout neat और organized दिखता है।
Types of alignment:

- **Left-aligned** – Text या image बाईं ओर align
- **Right-aligned** – Text दाईं ओर align
- **Center-aligned** – Middle में elements
- **Justified** – Text दोनों ओर aligned

Alignment से user को content पढ़ना और समझना आसान होता है।

### 4. Repetition (दोहराव)

Repetition का मतलब है — design elements जैसे color, font, icons, और layout pattern का consistent उपयोग।
यह consistency maintain करता है और website को professional look देता है।
Example: सभी headings के लिए same font और color scheme का उपयोग करना।
Repetition से branding भी मजबूत होती है।

### 5. Proximity (निकटता)

Proximity का अर्थ है — related items को एक साथ रखना और unrelated items को अलग रखना।
इससे user को content logically organized लगता है।
Example:

- Contact information (email, phone, address) को एक block में रखना।
- Menu items को group करना (Home, About, Services, Contact)।
  Proximity user navigation को intuitive बनाता है।

### 6. White Space (खाली स्थान)

White space या "Negative Space" वह area है जो elements के बीच खाली छोड़ा जाता है।
यह design को breathable और clean बनाता है।
White space readability और focus बढ़ाता है — इससे user important content पर ध्यान दे पाता है।
Overcrowded pages user experience को खराब कर देते हैं।

**7. Typography (टाइपोग्राफी)**

Typography का मतलब है — fonts का सही selection, size, spacing और style।
Good typography से text appealing और readable बनता है।
Tips:

- 2–3 font families से ज्यादा use न करें।
- Headings को bold और body text को simple रखें।
- Line spacing और letter spacing पर ध्यान दें।

**8. Color Scheme (रंग योजना)**

Color combination website की mood और branding को define करता है।
Color psychology के अनुसार:

- Blue – Trust और professionalism
- Green – Nature और growth
- Red – Energy और urgency
- Black – Elegance और power
  Use contrast colors for background and text ताकि readability बनी रहे।

**9. Consistency (संगतता)**

पूरी website में design consistent होना चाहिए — same navigation style, same button shapes, same color scheme।
Consistency से users को familiarity मिलती है और site navigation आसान लगता है।
Inconsistent design confusion पैदा करता है।

**10. Visual Hierarchy (दृश्य पदानुक्रम)**

Visual hierarchy का मतलब है — information को ऐसे present करना कि user की नज़र पहले सबसे important चीज़ पर जाए।
Methods to create hierarchy:

- Font size और color variations
- Headings और subheadings का proper use
- Images और call-to-action buttons को strategically रखना

# History of Internet (इंटरनेट का इतिहास)

## Introduction (परिचय)

Internet आज दुनिया की सबसे बड़ी communication network है जो करोड़ों computers और devices को आपस में जोड़ती है।
लेकिन इसका विकास अचानक नहीं हुआ — यह कई **दशकों के research, innovation, और collaboration** का परिणाम है।
Internet की शुरुआत military research से हुई और धीरे-धीरे यह academic, commercial, और social communication का global माध्यम बन गया।

## 1. Early Concept of Networking (नेटवर्किंग की प्रारंभिक अवधारणा)

1950s और 1960s में computer systems isolated (अलग-अलग) होते थे — उनमें आपसी data sharing नहीं होती थी।
इस समय **packet switching** का concept आया — जिसमें data को छोटे packets में तोड़कर destination तक भेजा जाता है।
यह concept later internet की foundation बना।

## 2. ARPANET (1960s–1970s)

Internet का जन्म **ARPANET (Advanced Research Projects Agency Network)** से हुआ।
इसे 1969 में **U.S. Department of Defense** ने develop किया ताकि universities और research centers आपस में connect हो सकें।

- 1969 में पहला ARPANET link **UCLA (University of California, Los Angeles)** और **Stanford Research Institute** के बीच स्थापित हुआ।
- शुरुआत में सिर्फ 4 computers connected थे।
- **1971** में email system विकसित हुआ, जिसने communication को आसान बना दिया।

ARPANET ने साबित किया कि long-distance computer communication संभव है।

## 3. TCP/IP Protocol (1980s)

ARPANET के success के बाद, scientists को एक **standard communication protocol** की ज़रूरत महसूस हुई।
**1983** में **TCP/IP (Transmission Control Protocol / Internet Protocol)** officially adopt किया गया।

- TCP/IP ने data transmission के लिए common rules set किए।
- इसने अलग-अलग networks को आपस में जोड़ने में मदद की।
  **1 जनवरी 1983** को इसे "Birthday of the Internet" माना जाता है।

**4. Domain Name System (DNS) – 1984**

शुरुआती दौर में websites को access करने के लिए numeric IP address याद रखना पड़ता था।
1984 में **Domain Name System (DNS)** शुरू हुआ, जिससे websites को readable names मिले (जैसे www.google.com)।
इससे internet का use आसान और user-friendly हो गया।

**5. Birth of World Wide Web (WWW) – 1990**

**Tim Berners-Lee**, एक British scientist, ने **1989–1990** में **World Wide Web (WWW)** invent किया।
WWW ने internet को एक interactive, graphical, और hyperlinked environment में बदल दिया।
उन्होंने तीन main technologies बनाई:

- **HTML (HyperText Markup Language)** – web pages बनाने के लिए
- **HTTP (HyperText Transfer Protocol)** – data transfer के लिए
- **URL (Uniform Resource Locator)** – web address के लिए

1991 में पहली website launch हुई — यह CERN (European Organization for Nuclear Research) से संबंधित थी।

**6. Commercialization of Internet (1990s)**

1990s में internet public के लिए खुल गया।

- 1993 में **Mosaic**, पहला graphical web browser आया।
- 1994 में **Netscape Navigator** popular हुआ।
- 1995 में **Microsoft Internet Explorer** launch हुआ।
- इसी समय online services जैसे **Yahoo (1994)**, **Amazon (1995)**, और **Google (1998)** शुरू हुए।
  1990s को **"Internet Boom Era"** कहा जाता है।

**7. Broadband and Wireless Revolution (2000s)**

2000s में internet तेज़ और accessible हुआ — dial-up से shift होकर broadband और Wi-Fi connections शुरू हुए।

- Social media platforms जैसे **Facebook (2004)**, **YouTube (2005)**, **Twitter (2006)** आए।
- Smartphones और mobile internet ने लोगों को "always connected" बना दिया।

इस दशक में internet communication, entertainment और business का मुख्य साधन बन गया।

**8. Cloud Computing and Smart Era (2010–Present)**

2010 के बाद से internet और intelligent बन गया है।

- **Cloud computing** ने online data storage और remote access संभव किया।
- **IoT (Internet of Things)** ने devices को interconnected किया।
- **AI (Artificial Intelligence)** और **Machine Learning** से internet personalized और efficient हुआ।
- 5G technology से high-speed connectivity की शुरुआत हुई।

आज internet केवल communication tool नहीं बल्कि **digital lifestyle** का हिस्सा है।

**9. Internet in India (भारत में इंटरनेट का विकास)**

- भारत में internet की शुरुआत **15 अगस्त 1995** को **VSNL (Videsh Sanchar Nigam Limited)** द्वारा हुई।
- 2000s में BSNL और private companies ने broadband service शुरू की।
- **Jio (2016)** के आगमन के बाद mobile internet भारत के हर कोने में पहुंच गया।
- आज भारत दुनिया के सबसे बड़े internet user base वाले देशों में से एक है।

## World Wide Web (विश्व व्यापी जाल)

**Introduction (परिचय)**

**World Wide Web (WWW)** या "Web" इंटरनेट पर उपलब्ध interconnected documents और resources का global system है।
यह users को **hypertext links** के माध्यम से एक page से दूसरे page तक जाने की सुविधा देता है।
WWW इंटरनेट का सबसे popular और useful हिस्सा है — जिसके माध्यम से हम websites, images, videos, news, और अन्य digital information access करते हैं।

World Wide Web को **Sir Tim Berners-Lee** ने 1989–1990 में **CERN (European Organization for Nuclear Research)** में develop किया था।
उन्होंने web को information sharing का आसान और universal तरीका बनाने के लिए design किया।

**Definition (परिभाषा)**

"World Wide Web is a system of interlinked hypertext documents accessed through the Internet using a web browser."

अर्थात — WWW इंटरनेट पर आपस में जुड़े documents (web pages) का एक ऐसा संग्रह है, जिसे web browser की मदद से access किया जाता है।

## 1. History and Development (इतिहास और विकास)

- **1989:** Tim Berners-Lee ने "Information Management: A Proposal" नामक document में web का विचार प्रस्तुत किया।
- **1990:** उन्होंने तीन मुख्य technologies develop कीं:
    1. **HTML (HyperText Markup Language)** – web page की structure के लिए
    2. **HTTP (HyperText Transfer Protocol)** – browser और server के बीच communication के लिए
    3. **URL (Uniform Resource Locator)** – हर resource के address के लिए
- **1991:** पहली website launch की गई, जो CERN के projects से संबंधित थी।
- **1993:** पहला graphical browser "Mosaic" आया, जिसने web को popular बना दिया।
- **1994:** Tim Berners-Lee ने "World Wide Web Consortium (W3C)" की स्थापना की ताकि web standards बनाए जा सकें।

## 2. Components of WWW (वेब के प्रमुख घटक)

*(a) Web Page (वेब पेज)*

एक single document जो HTML में लिखा जाता है और browser पर display होता है।
Example: Home page, About page, Contact page आदि।

*(b) Website (वेबसाइट)*

कई web pages का collection जो एक common domain name के तहत होता है।
Example: www.google.com, www.wikipedia.org

*(c) Web Browser (वेब ब्राउज़र)*

यह software है जो web pages को display करता है और user को web navigate करने देता है।
Common browsers: Google Chrome, Mozilla Firefox, Microsoft Edge, Safari

*(d) Web Server (वेब सर्वर)*

यह computer system होता है जो web pages और data को store करता है और client (browser) की request पर serve करता है।

*(e) URL (Uniform Resource Locator)*

यह web page या resource का unique address होता है, जैसे:
https://www.example.com/index.html

*(f) HTTP/HTTPS (HyperText Transfer Protocol / Secure)*

यह protocol browser और server के बीच data transfer करता है।
HTTPS इसका secure version है जो encryption प्रदान करता है।

### 3. Working of WWW (वेब का कार्य करने का तरीका)

1. User web browser में किसी web page का **URL** type करता है।
2. Browser **DNS (Domain Name System)** से उस domain का IP address प्राप्त करता है।
3. Browser **HTTP/HTTPS request** send करता है web server को।
4. Server requested web page (HTML file) browser को भेजता है।
5. Browser HTML, CSS, JavaScript को process करके page display करता है।

इस पूरे process को **Client-Server Communication** कहते हैं।

### 4. Features of World Wide Web (WWW की विशेषताएँ)

1. **Hyperlinking:** एक page से दूसरे page पर जाने की सुविधा hyperlinks के माध्यम से।
2. **Multimedia Support:** Text, image, video, audio और animation सभी को support करता है।
3. **Global Accessibility:** Internet connection से दुनिया के किसी भी कोने से access किया जा सकता है।
4. **Interactivity:** Forms, buttons और scripts के माध्यम से user interaction संभव है।
5. **Dynamic Content:** Websites user data के आधार पर content बदल सकती हैं।
6. **Cross-Platform Availability:** Mobile, tablet, desktop — हर device पर accessible।
7. **Searchability:** Search engines जैसे Google, Bing के माध्यम से information आसानी से खोजी जा सकती है।

### 5. Advantages of WWW (WWW के लाभ)

- जानकारी प्राप्त करने का सबसे तेज़ और आसान माध्यम
- Online education, shopping, entertainment, communication आदि में उपयोगी
- Business promotion और digital marketing के लिए powerful platform
- Global connectivity और collaboration को बढ़ावा देता है

### 6. Limitations of WWW (WWW की सीमाएँ)

- Security threats जैसे hacking, phishing, malware
- Data privacy का खतरा
- Fake information और misinformation की समस्या
- Internet connection पर निर्भरता

### 7. Evolution of WWW (वेब का विकास)

WWW समय के साथ कई phases से गुज़रा है:

| Web Version | Time Period | Description |
|---|---|---|
| **Web 1.0** | 1990s | Static web – केवल पढ़ने के लिए pages (Read-only) |
| **Web 2.0** | 2000s | Interactive web – users content बना और share कर सकते हैं (Read-Write) |
| **Web 3.0** | 2010s–Present | Semantic web – AI और personalization पर आधारित intelligent web |
| **Web 4.0 (Future)** | Upcoming | Fully intelligent, autonomous, and connected web (IoT और AI आधारित) |

## Why Create a Website (वेबसाइट क्यों बनाएं)

### Introduction (परिचय)

आज के digital युग में website किसी भी व्यक्ति, संस्था या व्यवसाय के लिए पहचान और जानकारी का सबसे प्रभावी माध्यम है।
Website केवल information दिखाने का platform नहीं, बल्कि यह communication, marketing, learning और interaction का global tool बन चुका है।

एक अच्छी website से user को information, services, और products तक आसान पहुंच मिलती है, वहीं owner को audience तक पहुँचने और अपनी credibility बढ़ाने का अवसर मिलता है।

### 1. To Establish Online Presence (ऑनलाइन पहचान बनाना)

Website आपके लिए **digital identity** का काम करती है।
आज हर व्यक्ति या संस्था जो अपने work को दुनिया तक पहुँचाना चाहती है, उसे web presence की आवश्यकता होती है।

- आपकी website 24×7 उपलब्ध रहती है।
- कोई भी व्यक्ति आपके बारे में जानकारी किसी भी समय प्राप्त कर सकता है।
- यह आपकी credibility (विश्वसनीयता) को बढ़ाती है।

**Example:** किसी college या institute की official website से students और parents को सभी जानकारी मिल जाती है।

**2. For Information Sharing (जानकारी साझा करने के लिए)**

Website information publish करने का सबसे आसान तरीका है।
आप announcements, notices, study materials, photos, videos, blogs आदि instantly share कर सकते हैं।
इससे communication effective और transparent बनता है।

**Example:** Government departments अपनी schemes और updates वेबसाइट पर publish करते हैं।

**3. For Business and Marketing (व्यवसाय और मार्केटिंग के लिए)**

Business के लिए website एक **digital showroom** की तरह होती है।

- Product details और services को showcase किया जा सकता है।
- Customers online order कर सकते हैं या contact कर सकते हैं।
- Digital marketing (SEO, ads, social media links) से business global reach पा सकता है।

**Example:** Amazon, Flipkart, Meesho जैसी companies अपनी website के माध्यम से लाखों ग्राहकों तक पहुँचती हैं।

**4. For Communication and Interaction (संचार और संपर्क के लिए)**

Websites users और organizations के बीच communication का माध्यम बनती हैं।

- Contact forms, live chat, feedback systems से users सीधा interaction कर सकते हैं।
- Students या clients queries भेज सकते हैं और timely response प्राप्त कर सकते हैं।

**Example:** College websites पर "Contact Us" या "Feedback" section इसी उद्देश्य से होता है।

**5. For Education and E-learning (शिक्षा के लिए)**

Education field में website का उपयोग तेजी से बढ़ा है।

- Colleges, schools और online academies अपने courses, study material और results publish करते हैं।
- Students assignments submit कर सकते हैं और online exams दे सकते हैं।
- Websites education को accessible और affordable बनाती हैं।

**Example:** SWAYAM, Coursera, edX जैसी learning websites।

**6. For E-Governance and Public Services (ई-शासन और जनसेवा के लिए)**

Government websites citizens को कई online सुविधाएँ देती हैं — जैसे certificates download, bill payments, registration, grievance redressal आदि।
इससे transparency और efficiency बढ़ती है।

**Example:** DigiLocker, UMANG, India.gov.in जैसी सरकारी websites।

**7. For Entertainment and Media (मनोरंजन और मीडिया के लिए)**

Movies, music, news, blogs, social media – सभी websites entertainment और information का source हैं।
People daily web platforms जैसे YouTube, Netflix, Hotstar, news portals आदि का उपयोग करते हैं।
Website content delivery का सबसे तेज़ माध्यम बन गई है।

**8. For Employment and Career (रोजगार और करियर के लिए)**

Job portals और professional websites career opportunities प्रदान करती हैं।

- Students resumes upload कर सकते हैं।
- Companies job vacancies publish कर सकती हैं।
- Freelancers अपने portfolio showcase कर सकते हैं।

**Example:** Naukri.com, LinkedIn, Freelancer.com आदि।

**9. For Personal Branding (व्यक्तिगत पहचान के लिए)**

Individuals अपने portfolio, blogs, research work या achievements दिखाने के लिए personal websites बनाते हैं।
यह students, teachers, artists या professionals के लिए career growth में मदद करता है।

**Example:** Portfolio websites – जैसे "www.yourname.com"

**10. For Revenue Generation (आय के लिए)**

Website से earning के कई तरीके हैं:

- Advertisements (Google AdSense)
- Affiliate marketing
- Online courses या eBooks
- Paid memberships या donations

कई लोग full-time bloggers या YouTubers बनकर websites से income generate करते हैं।

### 11. For Social and Community Purpose (सामाजिक कार्यों के लिए)

NGOs और social groups websites के ज़रिए awareness फैलाते हैं, donation campaigns चलाते हैं और public participation बढ़ाते हैं।
यह transparency और accountability को बढ़ाता है।

## Web Standards (वेब स्टैंडर्ड्स)

### Introduction (परिचय)

**Web Standards** ऐसे नियम और guidelines हैं जो web pages और applications को design और develop करने के लिए निर्धारित किए जाते हैं।
इन standards का उद्देश्य है कि websites **accessible, interoperable, और consistent** हों — यानी सभी browsers और devices पर ठीक से काम करें।

World Wide Web Consortium (W3C) मुख्य organization है जो web standards define करता है।
Standards adopt करने से website की **usability, performance, और SEO** बेहतर होती है।

### 1. Definition (परिभाषा)

"Web Standards are formal standards and best practices for designing and developing web content that ensure compatibility, accessibility, and usability across different browsers and devices."

सरल शब्दों में — यह नियम हैं जिनका पालन करके websites हर user और device के लिए functional और readable बनी रहें।

### 2. Importance of Web Standards (महत्व)

1. **Cross-Browser Compatibility (सभी browsers में काम करना)**
   Standards ensure करते हैं कि website Chrome, Firefox, Safari, Edge आदि सभी browsers में same तरीके से display हो।
2. **Accessibility (सभी users के लिए पहुँच योग्य)**
   Web standards यह सुनिश्चित करते हैं कि disabled या differently-abled users भी website access कर सकें।
3. **Consistency (संगति)**
   Standards follow करने से design और layout हर page पर consistent रहता है।
4. **Future-Proofing (भविष्य के लिए सुरक्षित)**
   Standard-compliant websites नए devices और browsers पर भी सही तरीके से काम करती हैं।

5. **SEO और Performance**
   Properly coded websites search engines में rank बेहतर पाती हैं और fast load होती हैं।

### 3. Types of Web Standards (वेब स्टैंडर्ड्स के प्रकार)

*(a) HTML Standards*

- Web page structure के लिए defined rules।
- Example: Proper use of headings ( to ), semantic tags (, , )।

*(b) CSS Standards*

- Style और layout define करने के लिए।
- Example: Use of classes, IDs, responsive layouts using media queries।

*(c) JavaScript Standards*

- Client-side scripting के लिए guidelines।
- Example: Avoid browser-specific code, follow ECMAScript standards।

*(d) Accessibility Standards (WCAG)*

- Website accessible हो सभी users के लिए।
- WCAG (Web Content Accessibility Guidelines) define करते हैं कैसे visually impaired या differently-abled users site use कर सकें।

*(e) Security Standards*

- Secure coding practices, HTTPS, data encryption।

### 4. Benefits of Following Web Standards (लाभ)

1. **Improved User Experience (बेहतर उपयोगकर्ता अनुभव)**
   Consistent design और fast loading pages users को बेहतर experience देते हैं।
2. **Reduced Development Time (विकास समय में कमी)**
   Standard guidelines follow करने से debugging और future maintenance आसान हो जाता है।
3. **Better Search Engine Rankings (बेहतर SEO)**
   Search engines standard-compliant websites को priority देते हैं।
4. **Device and Platform Independence (सभी डिवाइस और प्लेटफ़ॉर्म पर काम करना)**
   Desktop, tablet, mobile, या smart TV — हर device पर proper display।

5. **Professionalism and Credibility (विश्वसनीयता और professionalism)**
   Standard-compliant websites users और clients के लिए trustworthy दिखती हैं।

**5. Examples of Web Standards Organizations (संस्थाएँ)**

1. **W3C (World Wide Web Consortium)** – Main body for web standards.
2. **ECMA International** – JavaScript standards (ECMAScript).
3. **WHATWG (Web Hypertext Application Technology Working Group)** – HTML and DOM standards.
4. **IETF (Internet Engineering Task Force)** – Protocol standards like HTTP, HTTPS.

**6. Best Practices for Following Web Standards (सर्वोत्तम अभ्यास)**

- Use **semantic HTML** tags.
- Separate content (HTML), presentation (CSS), and behavior (JavaScript).
- Optimize images and media for fast loading.
- Ensure responsive design using media queries.
- Validate code using validators like **W3C Validator**.
- Ensure accessibility for differently-abled users (WCAG compliance).
- Use HTTPS for secure communication.

## Audience Requirements (ऑडियंस की आवश्यकताएँ)

**Introduction (परिचय)**

Website designing और development में **Audience Requirements** का मतलब है — users की ज़रूरतों, expectations और behavior को समझकर website बनाना।
Website तभी सफल होती है जब यह अपने **target audience** के लिए **useful, accessible, और engaging** हो।
Users की requirements को जानना design process का पहला और महत्वपूर्ण step है।

**1. Definition (परिभाषा)**

"Audience requirements are the specific needs, expectations, and preferences of the users for whom the website is being designed, which guide the design, content, and functionality of the website."

सरल शब्दों में — यह user-focused design का foundation है।

**2. Importance of Audience Requirements (महत्व)**

1. **Better User Experience (बेहतर उपयोगकर्ता अनुभव)**
   Users को ऐसा website मिले जो आसानी से navigate हो, fast load हो और उनकी जरूरत की जानकारी तुरंत मिले।
2. **Relevant Content (संबंधित सामग्री)**
   Users की preference समझकर relevant content provide करना आसान होता है।
3. **Increased Engagement (अधिक सहभागिता)**
   User-centered design से visitors website पर ज्यादा समय बिताते हैं और interact करते हैं।
4. **Business Goals Achieved (व्यवसायिक उद्देश्य प्राप्त होते हैं)**
   Target audience की सही understanding से website से lead generation, sales या conversions बढ़ सकते हैं।
5. **Accessibility and Inclusivity (पहुंच और समावेशिता)**
   Audience की diversity को ध्यान में रखकर design करना — जैसे age, abilities, language, और devices।

**3. Factors to Consider for Audience Requirements (ऑडियंस की आवश्यकताओं के लिए विचार करने वाले पहलू)**

*(a) Demographics (जनसांख्यिकी)*

- Age, gender, education level, occupation
- Example: Children-focused websites में colorful graphics और simple navigation होना चाहिए।

*(b) Technical Knowledge (तकनीकी ज्ञान)*

- Users की computer, mobile, और internet skills
- Example: Beginners के लिए simple interface, advanced users के लिए interactive features।

*(c) Purpose and Needs (उद्देश्य और जरूरतें)*

- Users website पर क्यों आते हैं — information, entertainment, shopping, learning
- Example: College website → students need notices, results, syllabus

*(d) Devices and Platforms (डिवाइस और प्लेटफ़ॉर्म)*

- Users desktop, tablet या mobile पर access करते हैं
- Mobile-friendly and responsive design जरूरी

*(e) Accessibility (सुविधा और पहुँच)*

- Users with disabilities (visual, hearing, motor)
- WCAG guidelines follow करना आवश्यक

*(f) Language and Culture (भाषा और संस्कृति)*

- Regional language support और culturally relevant content
- Example: Hindi-English bilingual interface for Indian audience

**4. Methods to Identify Audience Requirements (ऑडियंस की आवश्यकताओं को पहचानने के तरीके)**

1. **Surveys and Questionnaires (सर्वे और प्रश्नावली)**
   Users से उनकी preferences और expectations collect करना।
2. **Interviews (साक्षात्कार)**
   Target users से सीधे बात करके insight लेना।
3. **User Testing (उपयोगकर्ता परीक्षण)**
   Prototype या beta version users को दिखाकर feedback लेना।
4. **Analytics (वेबसाइट विश्लेषण)**
   Existing websites के traffic और behavior data analyze करना।
5. **Competitor Analysis (प्रतियोगी विश्लेषण)**
   Similar websites क्या offer कर रही हैं, और users उनकी response कैसी है।

**5. Examples of Audience Requirements (ऑडियंस आवश्यकताओं के उदाहरण)**

| Audience Type | Requirement Example |
|---|---|
| Students | Easy access to syllabus, notices, exam results |
| Business Users | Product information, online purchase options |
| General Public | Clear navigation, responsive design, contact info |
| Disabled Users | Screen reader support, text-to-speech, captions |

**6. Importance in Website Design Process (वेबसाइट डिजाइन प्रक्रिया में महत्व)**

1. **Content Planning:** Audience requirements determine what content should be on the website.
2. **Navigation Design:** Simple and intuitive navigation based on user behavior.
3. **Visual Design:** Colors, fonts, and layout chosen according to user preferences.
4. **Functionality:** Features like search, forms, chat, e-commerce tailored to audience needs.
5. **Accessibility:** Ensures website is inclusive for all users.

## 7. Benefits of Considering Audience Requirements (लाभ)

- Higher user satisfaction
- Increased retention and engagement
- Reduced bounce rate
- Better conversion rates
- Stronger brand reputation

**Introduction to HTML**

**HTML - HTML का पूरा नाम** – *Hyper Text Markup Language* है। यह एक **markup language** है जिसका उपयोग **web pages** और **web applications** बनाने के लिए किया जाता है। HTML किसी भी website का **basic structure** प्रदान करता है, जिस पर CSS और JavaScript जैसी technologies का प्रयोग करके उसे attractive और interactive बनाया जाता है।

## Meaning of Each Word:

1. **HyperText** – यह ऐसे text को दर्शाता है जिसमें **links (hyperlinks)** होते हैं, जिन पर click करके आप एक page से दूसरे page पर जा सकते हैं।
2. **Markup** – इसका अर्थ है कि HTML में **tags** का उपयोग करके text को structure दिया जाता है।
3. **Language** – क्योंकि यह browser को बताती है कि content को कैसे display करना है, इसलिए इसे language कहा जाता है।

## Features of HTML:

1. यह एक **platform-independent** language है।
2. HTML files का extension **.html** या **.htm** होता है।
3. इसमें **tags** और **attributes** का उपयोग होता है।
4. यह **text, image, video, link, table, form** आदि elements को web page में जोड़ने की सुविधा देता है।
5. इसे किसी भी **text editor** (जैसे Notepad, VS Code) में लिखा जा सकता है और किसी भी **web browser** में खोला जा सकता है।

## Basic Structure of an HTML Document:

```
<!DOCTYPE html>
<html>
<head>
  <title>My First Webpage</title>
</head>
<body>
  <h1>Welcome to HTML</h1>
  <p>This is my first web page.</p>
</body>
</html>
```

**HTML Document - HTML Document** वह file होती है जिसमें **HTML code** लिखा जाता है। यह document किसी भी **web page** की **complete structure और content** को define करता है। इसे web browser द्वारा पढ़कर **visual web page** के रूप में display किया जाता है।

HTML Document एक ऐसी **text file** होती है जो **Hyper Text Markup Language (HTML)** के **tags** और **elements** से मिलकर बनी होती है। इसमें हम headings, paragraphs, images, links, tables, forms आदि को define करते हैं ताकि browser उन्हें सही तरीके से दिखा सके।

हर HTML Document की शुरुआत **<!DOCTYPE html>** से होती है जो बताता है कि यह HTML5 document है। इसके बाद <html> tag के अंदर पूरा content लिखा जाता है। HTML Document को दो मुख्य भागों में बाँटा गया है —

1. **Head Section (<head>):**
   यह part browser को page की **information** देता है, जैसे — title, meta data, links to CSS/JS files आदि।
   **उदाहरण:**

```
<head>
  <title>My Web Page</title>
</head>
```

2. **Body Section (<body>):**
   यह part browser में **actual visible content** दिखाता है, जैसे headings, paragraphs, images, videos, tables आदि।
   उदाहरण:

```
<body>
    <h1>Welcome to My Website</h1>
    <p>This is an example of an HTML document.</p>
</body>
```

**Basic Structure of an HTML Document:**

```
<!DOCTYPE html>
<html>
<head>
  <title>Example Page</title>
</head>
<body>
  <h1>This is a Heading</h1>
  <p>This is a paragraph in an HTML document.</p>
</body>
</html>
```

**Features of an HTML Document:**

1. इसका extension **.html** या **.htm** होता है।
2. यह **text editor** (जैसे Notepad, VS Code) में बनाया जा सकता है।
3. यह **browser readable** होता है।
4. इसमें सभी elements **tags** के माध्यम से लिखे जाते हैं।
5. HTML Document web page का **structure और layout** define करता है।

**Basic Structure of an HTML Document** - **HTML (Hyper Text Markup Language)** किसी भी web page का **ढांचा (structure)** तैयार करने के लिए उपयोग की जाने वाली भाषा है। हर HTML file को **HTML Document** कहा जाता है। इस document में कुछ **standard tags** का प्रयोग किया जाता है जो browser को यह बताते हैं कि page का कौन-सा भाग क्या दर्शाता है।

## Basic Structure of an HTML Document:

एक HTML Document मुख्यतः 5 भागों में बँटा होता है —

1. <!DOCTYPE html>
2. <html> ... </html>
3. <head> ... </head>
4. <title> ... </title>
5. <body> ... </body>

### 1. <!DOCTYPE html> (Document Type Declaration):

यह line HTML document की **पहली line** होती है।
यह browser को बताती है कि file **HTML5** format में लिखी गई है।

Example:

```
<!DOCTYPE html>
```

- इसका कोई closing tag नहीं होता।
- यह browser को सही rendering mode चुनने में मदद करता है।

## 2. <html> Tag:

यह HTML document का **root element** है।
सभी अन्य tags <html> और </html> के बीच लिखे जाते हैं।

Example:

```
<html>
  ...content...
</html>
```

- इस tag में **lang** attribute दिया जा सकता है, जैसे —
  <html lang="en"> जो बताता है कि document English में है।

## 3. <head> Tag:

यह section browser को **page की information (metadata)** प्रदान करता है।
इसमें ऐसा content होता है जो सीधे web page पर **display नहीं होता**, लेकिन browser और search engines के लिए उपयोगी होता है।

Example:

```
<head>
   <title>My Web Page</title>
   <meta charset="UTF-8">
   <meta name="description" content="This is my first web page">
   <link rel="stylesheet" href="style.css">
</head>
```

**Head section के main elements:**

- <title> → Page का title set करता है (जो browser tab में दिखता है)
- <meta> → Extra information जैसे description, keywords
- <link> → CSS file से connection
- <script> → JavaScript file से link

## 4. <title> Tag:

यह tag <head> के अंदर होता है और page का **title** define करता है।
यह title browser की **title bar** या **tab** में दिखाई देता है।

Example:

```
<title>Welcome Page</title>
```

- Title SEO (Search Engine Optimization) के लिए भी महत्वपूर्ण होता है।

## 5. <body> Tag:

यह HTML document का सबसे महत्वपूर्ण भाग है क्योंकि इसमें वो content होता है जो user को web page पर दिखाई देता है।

Example:

```
<body>
   <h1>Welcome to My Website</h1>
   <p>This is an example of an HTML document structure.</p>
   <img src="image.jpg" alt="Example Image">
</body>
```

### Body section में आने वाले main elements:

- Headings (<h1> to <h6>)
- Paragraphs (<p>)
- Images (<img>)
- Links (<a>)
- Tables (<table>)
- Lists (<ul>, <ol>, <li>)
- Forms (<form>) आदि।

### Complete Example of an HTML Document:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>My First HTML Page</title>
</head>
<body>
   <h1>Welcome to HTML</h1>
   <p>This is the basic structure of an HTML document.</p>
</body>
</html>
```

**Creating an HTML Document - HTML Document** एक ऐसी file होती है जिसमें **HTML code** लिखा जाता है। इसे **web page बनाने** के लिए use किया जाता है। HTML document को बनाना बहुत आसान है — बस कुछ simple steps follow करने होते हैं।

नीचे step-by-step पूरी process को detail में समझाया गया है

Step 1: Open a Text Editor

सबसे पहले कोई भी **text editor** खोलें जिसमें आप code लिख सकें।
उदाहरण —

- Notepad (Windows)
- VS Code
- Sublime Text
- Notepad++ आदि।

Step 2: Write Basic HTML Structure

अब HTML का **basic structure** लिखें। हर HTML document कुछ main tags से बनता है — <!DOCTYPE html>, <html>, <head>, <title>, और <body>

Example:

```
<!DOCTYPE html>
<html>
<head>
   <title>My First Web Page</title>
</head>
<body>
   <h1>Welcome to HTML</h1>
   <p>This is my first HTML document.</p>
</body>
</html>
```

Step 3: Save the File

अब इस file को save करें —

1. **File → Save As** पर क्लिक करें।
2. File name दें, जैसे **index.html** या **mypage.htm**
3. "Save as type" में **All Files** चुनें।
4. Encoding को **UTF-8** रखें।

- File extension हमेशा **.html** या **.htm** होना चाहिए।

Step 4: Open the File in a Browser

अब saved file को किसी भी **web browser** (जैसे Chrome, Edge, Firefox) में खोलें।
इसके लिए आप file पर double-click करें या browser में **drag and drop** करें।

आपका web page अब browser में display होगा।

Step 5: Modify and Update

आप HTML document में content change करके उसे फिर से save कर सकते हैं।
Browser को refresh करने पर updated page दिखाई देगा।

**Example Output on Browser:**

| |
|---|
| Welcome to HTML<br>This is my first HTML document. |

**Important Points:**

1. HTML file का extension हमेशा .html या .htm होना चाहिए।
2. HTML document में सभी tags को सही ढंग से open और close करना ज़रूरी है।
3. Head section में page की information होती है और Body section में visible content।
4. HTML case-insensitive है (यानि <HTML> और <html> दोनों valid हैं)।

**Markup Tags in HTML - HTML (Hyper Text Markup Language)** में **Markup Tags** का उपयोग **web page की structure और formatting** को define करने के लिए किया जाता है।
HTML में "Markup" शब्द का अर्थ ही होता है — *marking up the text* यानी text को ऐसे तरीके से लिखना कि browser उसे समझ सके और सही format में दिखा सके।

## Definition:

**Markup Tags** वो special words या keywords होते हैं जो **angle brackets** < > के अंदर लिखे जाते हैं।
इनका उपयोग यह बताने के लिए किया जाता है कि content को web browser में **कैसे display करना है**।

## उदाहरण के लिए:

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

यहाँ <h1> और <p> दोनों **markup tags** हैं।

## Types of Markup Tags:

### 1. Container Tags (Paired Tags):

ये tags हमेशा **दो भागों** में होते हैं —
➤ **Opening tag** (<tagname>)
➤ **Closing tag** (</tagname>)
**Example:**

```
<p>This is a paragraph.</p>
<b>This is bold text.</b>
```

- Content दोनों tags के बीच लिखा जाता है।

### 2. Empty Tags (Singular or Self-closing Tags):

ये tags में **closing tag नहीं होता**, क्योंकि इनका काम अपने आप पूरा हो जाता है।
**Example:**

```
<br>   <!-- Line break -->
<hr>   <!-- Horizontal line -->
<img src="photo.jpg" alt="Image">  <!-- Image tag -->
```

- इन्हें **standalone tags** भी कहा जाता है।

**Commonly Used Markup Tags:**

| Tag | Description |
|---|---|
| <html> | HTML document की शुरुआत और अंत बताता है |
| <head> | Page की meta-information रखता है |
| <title> | Browser tab में title दिखाता है |
| <body> | Visible content रखता है |
| <h1> to <h6> | Headings दिखाने के लिए |
| <p> | Paragraph के लिए |
| <b> / <i> | Bold या italic text के लिए |
| <br> | Line break डालने के लिए |
| <a> | Hyperlink बनाने के लिए |
| <img> | Image दिखाने के लिए |

Example of HTML Markup Tags:

```
<!DOCTYPE html>
<html>
<head>
   <title>Markup Tag Example</title>
</head>
<body>
   <h1>Welcome to HTML</h1>
   <p>This is an example of markup tags.</p>
   <hr>
   <p><b>Thank You!</b></p>
</body>
</html>
```

- In this example, <h1>, <p>, <b>, <hr> सभी markup tags हैं जो page की layout और design तय करते हैं।

Features of Markup Tags:

1. Tags हमेशा **angle brackets < >** के अंदर लिखे जाते हैं।
2. Tags **case-insensitive** होते हैं (यानि <P> और <p> दोनों valid हैं)।
3. Tags browser को बताते हैं कि text या content को कैसे दिखाना है।
4. कुछ tags में **attributes** भी होते हैं जो extra information provide करते हैं (जैसे <img src="pic.jpg">)।
5. हर HTML document में कई markup tags का combination होता है।

**All Heading Tags of HTML** - **HTML (Hyper Text Markup Language)** में **Heading Tags** का उपयोग किसी web page पर **titles और headings** को दिखाने के लिए किया जाता है। Headings का मुख्य उद्देश्य है – **content को व्यवस्थित (organize)** करना और user व browser दोनों को यह बताना कि कौन-सा हिस्सा कितना महत्वपूर्ण है।

HTML में headings 6 levels की होती हैं — `<h1>` से `<h6>` तक।

## Meaning and Purpose of Heading Tags:

Web page बनाते समय हम अक्सर अलग-अलग sections या topics रखते हैं — जैसे main title, sub-title, sub-section आदि।
इन्हीं को visually और logically अलग दिखाने के लिए heading tags का use किया जाता है।

Example के तौर पर, एक web page के structure में —

- `<h1>` page का **main title** होता है।
- `<h2>` major **subheading** होती है।
- `<h3>`, `<h4>`, `<h5>`, `<h6>` further smaller headings को दर्शाते हैं।

### List of All HTML Heading Tags:

| Tag Name | Description | Visual Importance |
|---|---|---|
| <h1> | यह सबसे बड़ी और सबसे महत्वपूर्ण heading होती है। इसका प्रयोग page के main heading या title के लिए किया जाता है। | Highest |
| <h2> | यह <h1> के नीचे की heading होती है, जो major sections को दर्शाती है। | Second |
| <h3> | यह <h2> के sub-section को बताती है। इसका size थोड़ा छोटा होता है। | Medium |
| <h4> | यह <h3> से भी छोटा होता है और छोटे उप-विषयों के लिए प्रयोग होता है। | Small |
| <h5> | यह heading बहुत छोटी होती है और rarely use की जाती है। | Smaller |
| <h6> | यह सबसे छोटी heading होती है, बहुत कम importance वाले भागों के लिए। | Smallest |

## Syntax:

```
<h1>This is Heading 1</h1>
<h2>This is Heading 2</h2>
<h3>This is Heading 3</h3>
<h4>This is Heading 4</h4>
<h5>This is Heading 5</h5>
<h6>This is Heading 6</h6>
```

**Example (Complete HTML Document):**

```
<!DOCTYPE html>
<html>
<head>
  <title>HTML Heading Tags Example</title>
</head>
<body>
  <h1>Introduction to HTML</h1>
  <p>HTML stands for Hyper Text Markup Language. It is used to design web pages.</p>

  <h2>Basic Structure</h2>
  <p>An HTML document is divided into Head and Body sections.</p>

  <h3>Head Section</h3>
  <p>This section contains meta information and the title of the page.</p>

  <h3>Body Section</h3>
  <p>This section contains the actual content visible to the users.</p>

  <h4>Body Elements</h4>
  <p>Body may include headings, paragraphs, links, images, etc.</p>

  <h5>Example Subheading</h5>
  <p>This is a small heading for minor content.</p>

  <h6>Footer Note</h6>
  <p>This is the smallest heading, usually used for least important text.</p>
</body>
</html>
```

**Characteristics of Heading Tags:**

1. HTML में **6 heading levels** होते हैं – `<h1>` से `<h6>` तक।
2. Headings **automatically bold और block-level** होती हैं।
3. `<h1>` heading सबसे महत्वपूर्ण होती है और आमतौर पर **page title** के लिए उपयोग होती है।
4. `<h2>` से `<h6>` headings का size क्रमशः छोटा होता जाता है।
5. Headings को **SEO (Search Engine Optimization)** में भी बहुत महत्व दिया जाता है — search engines `<h1>` और `<h2>` headings को content के मुख्य keywords के रूप में पहचानते हैं।
6. Headings से **readability और structure** दोनों improve होते हैं।
7. एक web page में **केवल एक `<h1>` tag** होना चाहिए, ताकि page का main topic स्पष्ट रहे।

**Importance of Using Heading Tags:**

- यह page को **well-structured** और **organized** बनाते हैं।
- यह **readers** और **search engines** दोनों को content hierarchy समझने में मदद करते हैं।
- Heading tags web content को **accessible और scannable** बनाते हैं।
- सही heading order (hierarchy) से web page **professional और logical** दिखता है।

**Example of Proper Heading Hierarchy (Tree View):**

```
<h1>College Website</h1>
  <h2>Departments</h2>
    <h3>Computer Science</h3>
    <h3>Mechanical</h3>
  <h2>Facilities</h2>
    <h3>Library</h3>
    <h3>Hostel</h3>
  <h2>Contact Us</h2>
```

यह structure बताता है कि `<h1>` main topic है, `<h2>` उसके subtopics हैं, और `<h3>` sub-subtopics हैं।

**Paragraph Tags in HTML - HTML (Hyper Text Markup Language)** में **Paragraph Tag (`<p>`)** का उपयोग किसी भी web page में **text को paragraph में organize** करने के लिए किया जाता है। Paragraph tag एक **block-level element** है, जिसका मतलब है कि यह अपने आप एक **new line** में शुरू और खत्म होता है।

**Definition:**

**Paragraph Tag (`<p>`)** HTML का वह tag है जो text को **paragraph के रूप में** browser में display करता है।
इससे text **readable, organized और structured** दिखाई देता है।

**Syntax:**

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

**Key Features of Paragraph Tag:**

1.  एक **container tag** है, इसका **closing tag**  होता है।
2. Paragraph text को browser में **automatically line-break** के साथ display करता है।
3.  tag में हम **plain text, inline elements** (जैसे <b>, <i>, <a>) शामिल कर सकते हैं।
4. Paragraph tag **semantic HTML** का part है क्योंकि यह browser और search engines को बताता है कि यह content एक paragraph है।
5. Page layout में  का प्रयोग करके content को **spacing और alignment** दिया जा सकता है।

**Example of Paragraph Tag:**

```
<!DOCTYPE html>
<html>
<head>
   <title>Paragraph Tag Example</title>
</head>
<body>
   <h1>About HTML</h1>
   <p>HTML stands for Hyper Text Markup Language. It is used to create web pages.</p>
   <p>Paragraph tags are used to organize text into readable blocks. Each paragraph automatically starts on a new line.</p>
   <p><b>Note:</b> Paragraph tags can also include <i>inline elements</i> like bold, italic, or links.</p>
</body>
</html>
```

---

**Output (Browser Display):**

About HTML

HTML stands for Hyper Text Markup Language. It is used to create web pages.

Paragraph tags are used to organize text into readable blocks. Each paragraph automatically starts on a new line.

Note: Paragraph tags can also include inline elements like bold, italic, or links.

**Attributes of Paragraph Tag ( ):**

Paragraph tag में commonly कुछ attributes use किए जाते हैं:

1. **align** – Text alignment define करने के लिए

```
<p align="center">This text is centered.</p>
```

(Note: Modern HTML में CSS का use करना preferred है।)

2. **class** – Paragraph को CSS style देने के लिए

```
<p class="highlight">This is highlighted text.</p>
```

3. **id** – Paragraph को uniquely identify करने के लिए

```
<p id="intro">This is the introduction paragraph.</p>
```

**Important Points about** `<p>` **Tag:**

1. Paragraph tag **block-level element** है।
2. Browser automatically paragraph के start और end पर **space (margin)** add करता है।
3. Paragraph tag में **nested block-level elements** जैसे `<div>` या `<h1>` use नहीं किए जा सकते।
4. Inline elements जैसे `<b>`, `<i>`, `<a>`, `<span>` paragraph में safely use किए जा सकते हैं।

**Line Break Tags - HTML (Hyper Text Markup Language)** में **Line Break Tag (`<br>`)** का उपयोग text को **next line** में display कराने के लिए किया जाता है।
यह tag **inline element** है और browser में paragraph के अंदर text को **line-break** करने के लिए काम आता है।

**Definition:**

**Line Break Tag (`<br>`)** HTML का **self-closing tag** है जो text में **new line insert** करता है।

**Syntax:**

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

इस example में `<br>` के कारण text दूसरी line में start होगा।

## Key Features of <br> Tag:

1. <br> एक **empty (self-closing) tag** है, इसलिए इसका closing tag नहीं होता।
2. यह **block-level element नहीं**, बल्कि **inline element** है।
3. Paragraph या text के बीच **line-break insert** करने के लिए use किया जाता है।
4. Multiple <br> tags का use करके **multiple line breaks** create किए जा सकते हैं।
5. इसका use generally short lines, addresses, poems या lists में किया जाता है।

## Example of Line Break Tag:

```
<!DOCTYPE html>
<html>
<head>
   <title>Line Break Example</title>
</head>
<body>
   <h1>Line Break Example in HTML</h1>
   <p>This is the first line.<br>This is the second line.<br>This is the third line.</p>

   <p>Address:<br>123, Main Street<br>City: Bilaspur<br>State: Chhattisgarh</p>
</body>
</html>
```

## Output (Browser Display):

```
Line Break Example in HTML

This is the first line.
This is the second line.
This is the third line.

Address:
123, Main Street
City: Bilaspur
State: Chhattisgarh
```

## Important Points about <br> Tag:

1. <br> tag text को **next line** में move कर देता है।
2. यह **inline element** है, इसलिए text के बीच में use किया जाता है।
3. Paragraph या heading के अंदर <br> का use किया जा सकता है।
4. Multiple <br> tags जोड़कर extra vertical space create किया जा सकता है।
5. Modern HTML में **excessive <br> tag** का use avoid करना चाहिए, spacing के लिए **CSS margin/padding** preferred है।

# Elements of HTML

HTML Element - HTML Element web page का basic building block होता है। पूरे web page का structure, content, layout और behavior अलग-अलग HTML elements की मदद से तैयार किया जाता है।

एक HTML element सामान्यतः तीन भागों से मिलकर बनता है:

- Opening Tag
- Content
- Closing Tag

**General Syntax:**

```
<tagname>Content</tagname>
```

**Example:**

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

**यहाँ:**

-  → Opening tag (element की शुरुआत)
- This is a paragraph → Content
-  → Closing tag (element का अंत)

कुछ elements ऐसे भी होते हैं जिनमें content नहीं होता, इन्हें empty/void elements कहते हैं।

**HTML Elements का Importance**

- Web page का structure define करते हैं
- Content को readable और organized बनाते हैं
- Browser को बताते हैं कि कौन-सा content कैसे display करना है
- SEO और accessibility में मदद करते हैं
- CSS और JavaScript को apply करने का base बनाते हैं

**Types of HTML Elements**

**1.Container (Paired) Elements**

जिनमें opening और closing tag दोनों होते हैं।

**Examples:**

```
<h1>Heading</h1>

<p>Paragraph</p>

<div>Section</div>
```

**2.Empty / Void Elements**

जिनमें closing tag नहीं होता और ये self-closing होते हैं।

**Examples:**

- <br>
- <hr>
- <img>
- <input>
- <meta>
- <link>

**Major HTML Elements**

**1. Structural Elements**

**<html>**

यह पूरे HTML document का root element होता है। सभी elements इसी के अंदर लिखे जाते हैं।

```
<html>

  ...

</html>
```

**<head>**

Web page की background information रखता है, जो user को directly दिखाई नहीं देती।

Includes:

<title>

<meta>

<link>

<script>

**<title>**

Browser tab में दिखने वाला page का नाम।

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

**<body>**

Web page का वह भाग जो user को दिखाई देता है।

```
<body>
  Visible content
</body>
```

**2. Text Content & Formatting Elements**

**Headings <h1> to <h6>**

Page के headings define करते हैं।<h1> सबसे important और <h6> सबसे छोटा heading होता है।

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

**<p> (Paragraph)**

Text को paragraph के रूप में लिखने के लिए।

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

**<br> (Line Break)**

Line को next line में ले जाता है।

**<hr> (Horizontal Rule)**

Content के बीच separation दिखाने के लिए horizontal line।

**Text Formatting Tags**

| Tag | Purpose |
|---|---|
| <b> | Bold text |
|  | Important text |
| <i> | Italic |
|  | Emphasized text |
| <u> | Underline |
| <mark> | Highlight |
| <small> | Small text |

**3. List Elements**

**<ul> (Unordered List)**

Bullet points वाली list।

**<ol> (Ordered List)**

Numbered list।

**<li> (List Item)**

List के items।

**4. Link & Media Elements**

**<a> (Anchor Tag)**

Hyperlink create करने के लिए।

```
<a href="https://example.com">Visit</a>
```

**<img>**

Image display करने के लिए।

```
<img src="photo.jpg" alt="image">
```

alt attribute accessibility के लिए बहुत जरूरी है।

**<audio> और <video>**

Multimedia content embed करने के लिए।

**5. Table Elements**

Data को rows और columns में show करने के लिए।

| Tag | Work |
|---|---|
| <table> | Table start |
| <tr> | Row |
| <th> | Header cell |
| <td> | Data cell |

Tables reports, marksheets और records में बहुत useful होते हैं।

**6. Form Elements**

User से data input लेने के लिए।

Common Form Elements:

<form>

<input>

<textarea>

<select>

<option>

<button>

Forms login, registration, feedback आदि के लिए use होते हैं।

### 7. Semantic HTML Elements

ये elements content का meaning बताते हैं, सिर्फ layout नहीं।

| Element | Meaning |
|---|---|
| <header> | Page header |
| <nav> | Navigation |
| <section> | Section |
| <article> | Independent content |
| <aside> | Side content |
| <footer> | Footer |

Semantic elements SEO और accessibility improve करते हैं।

### 8. Generic Container Elements

** **

Block-level container, layout design में use होता है।

****

Inline container, small text styling के लिए।

### 9. Meta & Resource Elements

**<meta>**

Page information जैसे charset, viewport।

**<link>**

External CSS जोड़ने के लिए।

**<script>**

JavaScript जोड़ने के लिए।

**Text-Related HTML Tags -** Text-related HTML tags वे tags होते हैं जिनका उपयोग web page पर text को लिखने, format करने, emphasize करने और meaningful बनाने के लिए किया जाता है।
इन tags की मदद से हम text का size, style, importance, alignment और meaning define करते हैं।

ये tags content presentation के साथ-साथ SEO और accessibility में भी important role निभाते हैं।

### 1. Heading Tags (<h1> to <h6>)

Heading tags का उपयोग headings और sub-headings दिखाने के लिए किया जाता है।

```
<h1>Main Heading</h1>

<h2>Sub Heading</h2>
```

**Features:**

<h1> सबसे बड़ा और सबसे important heading होता है

<h6> सबसे छोटा heading होता है

Page पर hierarchy (structure) बनती है

एक page पर ideally एक ही <h1> होना चाहिए।

### 2. Paragraph Tag (<p>)

Text को paragraph के रूप में लिखने के लिए use होता है।

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

**Characteristics:**

- Automatically new line से start होता है
- Paragraph के बीच default margin होती है
- Long text को readable बनाता है

**3. Line Break Tag (<br>)**

Text को next line में ले जाने के लिए use होता है।

```
Address:<br>
Bilaspur, Chhattisgarh
```

यह empty tag है, closing tag नहीं होता।

**4. Horizontal Rule Tag (<hr>)**

Content sections के बीच horizontal line draw करता है।

```
<hr>
```

**Use:**

- Topics को separate करने के लिए
- Visual break देने के लिए

**5. Bold Text Tags (<b> और )**

**<b> – Bold**

```
<b>Bold Text</b>
```

केवल visual bold देता है, कोई semantic meaning नहीं।

** – Important Text**

```
<strong>Important Text</strong>
```

Text की importance बताता है (semantic meaning)।

SEO और screen readers  को prefer करते हैं।

**6. Italic Text Tags (<i> और )**

**<i> – Italic**

```
<i>Italic Text</i>
```

Visual styling के लिए।

** – Emphasis**

```
<em>Emphasized Text</em>
```

Text पर emphasis डालता है, semantic meaning देता है।

**7. Underline Tag (<u>)**

Text को underline करने के लिए।

```
<u>Underlined Text</u>
```

Mostly avoid किया जाता है क्योंकि link जैसा दिख सकता है।

**8. Mark Tag (<mark>)**

Text को highlight करने के लिए।

```
<mark>Important</mark>
```

**Use case:**

- Search result highlight
- Important words दिखाने के लिए

**9. Small Tag (<small>)**

Text को small size में दिखाने के लिए।

```
<small>Terms and conditions apply</small>
```

**Use:**

- Copyright
- Disclaimer

**10. Superscript Tag (<sup>)**

Text को upper side में दिखाता है।

```
x<sup>2</sup>
```

**Use:**

- Mathematical power
- Footnotes

**11. Subscript Tag (<sub>)**

Text को lower side में दिखाता है।

```
H<sub>2</sub>O
```

**Use:**

Chemical formulas

**12. Preformatted Text (<pre>)**

Text को as-it-is format में दिखाता है।

```
<pre>
Name   Age
Ram    20
</pre>
```

**Features:**

- Spaces और line breaks preserve रहते हैं
- Code snippets के लिए useful

**13. Code Tag (<code>)**

Programming code दिखाने के लिए।

```
<code>print("Hello")</code>
```

Often <pre> के साथ use किया जाता है।

**14. Quotation Tags**

**<q> – Short Quotation**

```
<q>Knowledge is power</q>
```

**<blockquote> – Long Quotation**

```
<blockquote>
This is a long quotation.
</blockquote>
```

**15. Abbreviation Tag (<abbr>)**

Abbreviation का full form दिखाने के लिए।

```
<abbr title="World Health Organization">WHO</abbr>
```

Accessibility improve करता है।

**16. Citation Tag (<cite>)**

Book, article या author को cite करने के लिए।

```
<cite>Bhagavad Gita</cite>
```

**17. Span Tag ()**

Inline container, text styling के लिए।

```
<span style="color:red;">Red Text</span>
```

**List HTML Tags -** List HTML tags का उपयोग web page पर items को structured, ordered और readable format में show करने के लिए किया जाता है।
Lists content को systematic तरीके से present करती हैं, जिससे user को information समझने में आसानी होती है।

**HTML में lists का उपयोग:**

- Menu बनाने में
- Instructions दिखाने में
- Steps, rules, features आदि show करने में
- Navigation bars और forms में

**Types of Lists in HTML**

HTML में मुख्य रूप से तीन प्रकार की lists होती हैं:

1. Unordered List
2. Ordered List
3. Description (Definition) List

### 1. Unordered List (<ul>)

Unordered List में items किसी specific order में नहीं होते। Items के आगे bullets (•, ○, ■) दिखाई देते हैं।

**Syntax:**

```
<ul>
  <li>Item 1</li>
  <li>Item 2</li>
</ul>
```

**Example:**

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

**Characteristics:**

- Order important नहीं होता
  Bullets default होते हैं
  Menus और feature lists में उपयोगी

### 2. Ordered List (<ol>)

Ordered List में items का order (क्रम) important होता है। Items numbers, letters या roman numerals से show होते हैं।

**Syntax:**

```
<ol>
  <li>Step 1</li>
  <li>Step 2</li>
</ol>
```

**Example:**

```
<ol>
  <li>Turn on computer</li>
  <li>Open browser</li>
</ol>
```

**List Item Tag (<li>)**

<li> tag list का individual item define करता है। यह <ul> और <ol> दोनों के अंदर use होता है।

```
<li>List Item</li>
```

**Important Points:**

- <li> अकेले use नहीं किया जा सकता
- हर item <li> में होना जरूरी

**3. Description / Definition List (<dl>)**

Description List का उपयोग term और उसकी description दिखाने के लिए किया जाता है।

**Related Tags:**

| Tag | Meaning |
|---|---|
| <dl> | Description list |
| <dt> | Description term |
| <dd> | Description data |

**Syntax:**

```
<dl>
  <dt>HTML</dt>
  <dd>HyperText Markup Language</dd>
</dl>
```

**Use Cases:**

Dictionary
Glossary

FAQs
Definitions

**Nested Lists (List के अंदर List)**

एक list के अंदर दूसरी list को nested list कहते हैं।

```
<ul>
  <li>Fruits
    <ul>
      <li>Apple</li>
      <li>Mango</li>
    </ul>
  </li>
</ul>
```

Nested lists menus और sub-menus में बहुत उपयोगी होती हैं।

**Table Tags -** HTML Table tags का उपयोग web page पर data को rows और columns के form में structured तरीके से display करने के लिए किया जाता है।
Tables खासतौर पर records, reports, marksheets, timetables, price lists, result sheets आदि दिखाने के लिए उपयोगी होते हैं।

HTML table data को tabular form में organize करती है जिससे information clear और readable बनती है।

**Basic Structure of an HTML Table**

```
<table>
  <tr>
    <th>Header</th>
    <th>Header</th>
  </tr>
  <tr>
```

```
    <td>Data</td>
    <td>Data</td>
  </tr>
</table>
```

Table हमेशा <table> tag से start होती है और उसके अंदर rows, headers और data cells होते हैं।

**Table-Related HTML Tags**

**1. <table> Tag**

<table> tag पूरे table को define करता है। सभी table-related tags इसी के अंदर लिखे जाते हैं।

```
<table>
  ...
</table>
```

**Role:**

- Table structure create करता है
- Rows और columns को hold करता है

**2. <tr> – Table Row**

<tr> tag table की एक complete row को represent करता है।

```
<tr>
  ...
</tr>
```

**Features:**

- Horizontal row बनाता है
- Header row और data row दोनों के लिए use होता है

### 3. <th> – Table Header Cell

<th> tag table के heading cells के लिए use होता है।

```
<th>Name</th>
```

**Characteristics:**

- Text by default bold होता है
- Content automatically center aligned होता है
- Column या row का title बताता है

### 4. <td> – Table Data Cell

<td> tag table के actual data को show करता है।

```
<td>Ram</td>
```

**Features:**

- Normal text
- Left aligned by default
- Data entry के लिए use होता है

### 5. <caption> – Table Caption

Table का title या heading define करता है।

```
<caption>Student Record</caption>
```

**Use:**

- Table का purpose बताता है
- Screen readers के लिए useful

<caption> हमेशा <table> के तुरंत बाद लिखा जाता है।

### 6. <thead> – Table Head Section

Table का header section define करता है।

```
<thead>
  <tr>
    <th>Name</th>
```

```
      <th>Age</th>
    </tr>
  </thead>
```

**Importance:**

- Table structure clear बनाता है
- Styling और scripting आसान बनाता है

### 7. <tbody> – Table Body Section

Table का main data part।

```
  <tbody>
    <tr>
      <td>Ram</td>
      <td>20</td>
    </tr>
  </tbody>
```

### 8. <tfoot> – Table Footer Section

Table का footer part define करता है।

```
  <tfoot>
    <tr>
      <td>Total</td>
      <td>100</td>
    </tr>
  </tfoot>
```

**Use:**

Totals, summary, remarks show करने के लिए

**9. rowspan Attribute**

एक cell को multiple rows में merge करने के लिए।

```
<td rowspan="2">Total</td>
```

**10.** colspan **Attribute**

एक cell को multiple columns में merge करने के लिए।

```
<td colspan="3">Total Marks</td>
```

**Complete Table Example**

```
<table border="1">
  <caption>Student Marks</caption>
  <thead>
    <tr>
      <th>Name</th>
      <th>Math</th>
      <th>Science</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Ram</td>
      <td>80</td>
      <td>75</td>
    </tr>
    <tr>
      <td>Shyam</td>
      <td>70</td>
      <td>85</td>
```

```
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td colspan="3">End of Record</td>
    </tr>
  </tfoot>
</table>
```

**Frame -** HTML Frames का उपयोग browser window को multiple sections (frames) में divide करने के लिए किया जाता था, जहाँ हर section में अलग-अलग HTML document load किया जा सकता था।

**Frames की मदद से:**

- Page का एक हिस्सा static रखा जा सकता था (menu)
- दूसरा हिस्सा dynamic रखा जा सकता था (content)
- Multiple pages को एक ही window में दिखाया जा सकता था

**Important Note:**
HTML5 में frames deprecated (obsolete) हो चुके हैं|

**Frame-Related Tags**

HTML frames से जुड़े मुख्य tags ये हैं:

<frameset>
<frame>
<noframes>
<iframe> (modern & important)

**1. <frameset> Tag**

<frameset> tag <body> की जगह use होता है और browser window को rows या columns में divide करता है।

**Syntax:**

```
<frameset rows="50%,50%">
  ...
</frameset>
```

**Attributes:**

| Attribute | Description |
|---|---|
| rows | Horizontal division |
| cols | Vertical division |

**Example:**

```
<frameset cols="30%,70%">
  <frame src="menu.html">
  <frame src="content.html">
</frameset>
```

**Explanation:**

- Screen दो parts में divide होगी
- Left में menu.html
- Right में content.html

### 2. <frame> Tag

<frame> tag individual frame define करता है और बताता है कि कौन-सा HTML page load होगा।

**Syntax:**

```
<frame src="page.html">
```

**Common Attributes:**

| Attribute | Use |
|---|---|
| src | HTML file load करने के लिए |
| name | Frame को name देना |
| scrolling | yes / no |
| noresize | Resize रोकने के लिए |

**Example:**

```
<frame src="header.html" name="top">
```

### 3. <noframes> Tag

<noframes> tag उन browsers के लिए होता है जो frames support नहीं करते।

**Syntax:**

```
<noframes>
  <body>Your browser does not support frames</body>
</noframes>
```

**Purpose:**

- Compatibility
- Fallback content provide करना

### 4. Nested Frames

Frameset के अंदर दूसरा frameset use करना nested frames कहलाता है।

```
<frameset rows="20%,80%">
  <frame src="top.html">
  <frameset cols="30%,70%">
    <frame src="left.html">
    <frame src="right.html">
  </frameset>
</frameset>
```

### 5. <iframe> – Inline Frame

<iframe> tag का उपयोग एक HTML page के अंदर दूसरा HTML page embed करने के लिए किया जाता है।

यह HTML5 में valid और widely used है।

**Syntax:**

```
<iframe src="page.html"></iframe>
```

**Common Uses of <iframe>:**

- Google Maps embed
- YouTube video
- External websites
- Advertisements

**Important Attributes of <iframe>:**

| Attribute | Description |
|---|---|
| src | Page URL |
| width | Width |
| height | Height |
| frameborder | Border (deprecated) |
| allowfullscreen | Fullscreen |
| loading | lazy loading |

**Example:**

```
<iframe
  src="https://www.youtube.com/embed/xyz"
  width="400"
  height="300"
  allowfullscreen>
</iframe>
```

**Hyperlink Tags**

**Hyperlink** - Hyperlink वह clickable text, image या object होता है जिसकी मदद से user एक web page से दूसरे web page पर, एक website से दूसरी website पर, उसी page के किसी specific section पर, या किसी file (PDF, image, mail) पर navigate कर सकता है।

Hyperlinks ही web को **"**interconnected**"** बनाते हैं।

**Hyperlink बनाने वाला मुख्य Tag – <a> (Anchor Tag)**

HTML में hyperlink बनाने के लिए <a> (anchor) tag का उपयोग किया जाता है।

**Basic Syntax:**

```
<a href="url">Link Text</a>
```

**Example:**

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

**यहाँ:**

- <a> → Anchor tag
- href → Hypertext Reference (destination URL)
- Link Text → User को दिखाई देने वाला text

**Types of Hyperlinks**

**1. External Hyperlink**

किसी दूसरी website पर जाने के लिए।

```
<a href="https://www.wikipedia.org">Wikipedia</a>
```

**Use:**

- Other websites
- Reference links

**2. Internal Hyperlink**

उसी website के दूसरे page पर जाने के लिए।

```
<a href="about.html">About Us</a>
```

**Use:**

- Website navigation
- Menu system

**3. Intra-page / Bookmark Link**

उसी page के किसी specific section पर जाने के लिए।

Step 1: ID define करें

```
<h2 id="contact">Contact Us</h2>
```

Step 2: Link बनाएं

```
<a href="#contact">Go to Contact</a>
```

**4. Email Link (mailto)**

Email client open करने के लिए।

```
<a href="mailto:info@example.com">Send Email</a>
```

User के system में email client configured होना चाहिए।

**5. Telephone Link (tel)**

Mobile devices पर direct call करने के लिए।

```
<a href="tel:+919876543210">Call Us</a>
```

**6. File Download Link**

PDF, image, document download करने के लिए।

```
<a href="file.pdf" download>Download PDF</a>
```

**7. Image as Hyperlink**

Image पर click करने से link open होता है।

```
<a href="home.html">
  <img src="logo.png" alt="Logo">
</a>
```

**Important Attributes of Anchor Tag (<a>)**

**1. href (Mandatory)**

Link का destination define करता है।

**2. target**

Link कहाँ खुलेगी, यह decide करता है।

| Value | Meaning |
|---|---|
| _self | Same tab (default) |
| _blank | New tab |
| _parent | Parent frame |
| _top | Full window |

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

**3. title**

Hover करने पर tooltip दिखाता है।

```
<a href="home.html" title="Go to Home Page">Home</a>
```

**4. download**

File download force करता है।

**5. rel**

Security और SEO के लिए use होता है।

```
<a href="https://example.com" target="_blank" rel="noopener noreferrer">
```

**Image Tags -** HTML में images web page को visually attractive, informative और interactive बनाती हैं।
Images का उपयोग logos, photos, diagrams, banners, icons आदि दिखाने के लिए किया जाता है।

HTML में image display करने के लिए <img> tag का उपयोग किया जाता है।

**<img> Tag** - <img> एक empty (void) tag है, यानी इसका कोई closing tag नहीं होता। यह tag image को web page पर embed करता है।

**Basic Syntax:**

```
<img src="image.jpg" alt="Image description">
```

**Important Attributes of** <img> **Tag**

**1. src (Source) Attribute**

यह image का path या URL specify करता है।

```
<img src="photo.jpg">
```

**Types of paths:**

Relative path → images/pic.jpg

Absolute path → https://example.com/pic.jpg

src के बिना image display नहीं होगी।

**2. alt (Alternate Text)**

Image load न होने पर दिखने वाला text।
यह accessibility और SEO के लिए बहुत important है।

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

**Importance:**

- Screen readers इसे पढ़ते हैं
- Slow internet पर helpful
- Search engines image को understand करते हैं

### 3. width और height

Image का size define करते हैं।

```
<img src="pic.jpg" width="300" height="200">
```

### 4. title Attribute

Mouse hover करने पर tooltip दिखाता है।

```
<img src="pic.jpg" title="Nature Image">
```

### Image as a Hyperlink

Image को clickable link बनाने के लिए <a> tag के अंदर रखा जाता है।

```
<a href="home.html">
  <img src="logo.png" alt="Home">
</a>
```

### Image Formats Supported

| Format | Use |
|---|---|
| JPG / JPEG | Photos |
| PNG | Transparent images |
| GIF | Animations |
| SVG | Scalable graphics |
| WebP | Optimized images |

**Multimedia Tags**

**Multimedia -** Multimedia का अर्थ है एक से अधिक media का उपयोग, जैसे:

- Audio (sound)
- Video
- Images
- Animations

HTML में multimedia tags का उपयोग web page को interactive, engaging और user-friendly बनाने के लिए किया जाता है।

**Major Multimedia Tags in HTML**

HTML में मुख्य multimedia tags ये हैं:

1. <audio>
2. <video>
3. <source>
4. <track>

**1. <audio> Tag (Audio Tag)**

<audio> tag का उपयोग web page पर audio files (music, voice, sound effects) play करने के लिए किया जाता है।

**Basic Syntax:**

```
<audio controls>
  <source src="song.mp3" type="audio/mpeg">
</audio>
```

**Important Attributes of <audio>:**

| Attribute | Description |
|---|---|
| controls | Play, pause, volume |
| autoplay | Automatically play |
| loop | Repeat audio |
| muted | Mute sound |
| preload | Loading behavior |

<audio controls autoplay loop>

**2. <video> Tag (Video Tag)**

<video> tag का उपयोग video files को web page पर play करने के लिए किया जाता है।

**Basic Syntax:**

```
<video width="400" controls>
  <source src="movie.mp4" type="video/mp4">
</video>
```

**Important Attributes of <video>:**

| Attribute | Purpose |
|---|---|
| controls | Video controls |
| autoplay | Auto play |
| loop | Repeat |
| muted | Silent start |
| poster | Thumbnail image |
| width/height | Size |

```
<video controls poster="thumb.jpg">
```

**3. <source> Tag**

<source> tag audio और video tags के अंदर use होता है और media file का path और type बताता है।

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

Multiple formats देने से browser compatibility बढ़ती है।

**4. <track> Tag (Subtitles & Captions)**

<track> tag का उपयोग subtitles, captions, descriptions देने के लिए किया जाता है।

```
<track src="subtitles.vtt" kind="subtitles" srclang="en">
```

**Uses:**

- Accessibility (hearing impaired users)
- Multiple languages

**Advantages of Multimedia Tags**

- No plugin required
- Mobile friendly
- Better performance
- SEO friendly
- Accessibility support

**HTML Form and Control Tags**

**Form -** HTML Form का उपयोग user से input / data collect करने के लिए किया जाता है। यह data बाद में server, database या backend program को भेजा जाता है।

Examples of data:

- Name, Address
- Login details
- Feedback, Registration
- Online applications

**<form> Tag**

<form> tag form का container होता है। सभी form controls इसी के अंदर आते हैं।

Syntax:

```
<form action="submit.php" method="post">
  form elements
</form>
```

**Important Attributes of <form>:**

| Attribute | Description |
|---|---|
| action | Server file / URL |
| method | GET / POST |
| target | Response location |
| autocomplete | On / Off |
| novalidate | Disable validation |

**action attribute** - action attribute यह बताता है कि form submit होने के बाद data किस server file या URL को भेजा जाएगा। यानि form data की destination decide करता है।

**Syntax:**

```
<form action="submit.php">
```

**Explanation:**

जब user Submit button पर click करता है तो form में भरा हुआ सारा data action में दिए गए server page / script को चला जाता है

**Example:**

```
<form action="login.py" method="post">
```

यहाँ form data login.py file को भेजा जाएगा।

Special Cases:

| **Action Value** | **Meaning** |
|---|---|
| action="" | Same page पर data submit |
| action="#" | Page reload |
| action="https://site.com" | External server |

**method attribute**- method attribute यह बताता है कि form data किस तरीके (HTTP method) से भेजा जाएगा।

**Syntax:**

```
<form method="get">
```

या

```
<form method="post">
```

**Types of Method**

**(a) GET Method**- GET method में form data URL के साथ attach होकर server को भेजा जाता है।

**Example:**

```
<form action="search.php" method="get">
```

**Characteristics:**

- Data URL में visible
- Limited data size
- Fast
- Bookmark possible
- Less secure

**(b) POST Method** - POST method में form data HTTP request body में भेजा जाता है, URL में दिखाई नहीं देता।

**Example:**

```
<form action="register.php" method="post">
```

**Characteristics:**

- Data hidden
- Large data transfer
- More secure
- File upload supported
- Bookmark possible नहीं

**Difference between GET and POST Method**

| Basis of Comparison | GET Method | POST Method |
|---|---|---|
| Definition | GET method में form data को URL के साथ attach करके server को भेजा जाता है। | POST method में form data को HTTP request body के अंदर भेजा जाता है। |
| Data Visibility | Data URL में visible रहता है। | Data URL में visible नहीं रहता। |
| Security | Security कम होती है क्योंकि data दिखाई देता है। | Security ज्यादा होती है क्योंकि data hidden रहता है। |
| Data Length / Size | Data size limited होता है (URL length limitation)। | Data size पर कोई practical limit नहीं होती। |

| | | |
|---|---|---|
| Usage Purpose | Data retrieve करने के लिए use होता है। | Data submit / save / update करने के लिए use होता है। |
| Sensitive Data | Password, bank details जैसे data के लिए suitable नहीं। | Sensitive data के लिए suitable। |
| Speed | GET method fast होता है क्योंकि simple request होती है। | POST method थोड़ा slower होता है। |
| Bookmarking | GET request को bookmark किया जा सकता है। | POST request को bookmark नहीं किया जा सकता। |
| Caching | GET request cache हो सकती है। | POST request cache नहीं होती। |
| File Upload | File upload possible नहीं। | File upload possible है। |
| Encoding Type | Data URL encoding में भेजा जाता है। | Data body में encoding के साथ भेजा जाता है। |
| Re-submit Issue | Page refresh करने पर re-submit का issue नहीं। | Page refresh पर re-submit warning आ सकती है। |
| Use in Search | Search engines में query भेजने के लिए suitable। | Search operation के लिए generally use नहीं होता। |
| Example URL | login.php?user=ram&pwd=123 | URL में कोई data नहीं दिखता। |
| HTML Syntax | <form method="get"> | <form method="post"> |
| Server Side Visibility | Query string से data access किया जाता है। | Request body से data access किया जाता है। |
| Data Storage Risk | Browser history में save हो सकता है। | Browser history में save नहीं होता। |

**Form Control Tags (Input Elements)**

**1. <input> Tag**

<input> सबसे important और versatile form control है। इसका behavior type attribute से तय होता है।

**Syntax:**

```
<input type="text" name="username">
```

**Common Input Types:**

| Type | Use |
|---|---|
| text | Single line text |
| password | Hidden text |
| email | Email input |
| number | Numeric value |
| radio | Single selection |
| checkbox | Multiple selection |
| date | Date picker |
| file | File upload |
| submit | Submit button |
| reset | Clear form |
| button | Custom button |
| hidden | Hidden data |

**Example:**

```
<input type="radio" name="gender"> Male
<input type="radio" name="gender"> Female
```

**2. <label> Tag**

<label> tag form control को identify करने के लिए use होता है।

```
<label for="name">Name:</label>
<input type="text" id="name">
```

Improves accessibility and usability.

**3. <textarea> Tag**

Multi-line text input के लिए।

```
<textarea rows="4" cols="30"></textarea>
```

**Used for:**

- Address
- Feedback
- Comments

**4. <select> Tag (Dropdown List)**

Dropdown list बनाने के लिए।

```
<select>
  <option>India</option>
  <option>USA</option>
</select>
```

**<option> Tag**

Dropdown item define करता है।

**5. <button> Tag**

Clickable button create करता है।

```
<button type="submit">Submit</button>
```

**Button Types:**

- submit
- reset
- button

**6. <fieldset> Tag**

Related form controls को group करने के लिए।

```
<fieldset>
  <legend>Personal Info</legend>
</fieldset>
```

**7. <legend> Tag**

<fieldset> का title देता है।

### 8. <datalist> Tag

Input के लिए suggestion list देता है।

```
<input list="city">
<datalist id="city">
  <option value="Raipur">
</datalist>
```

### Form Validation Attributes

| Attribute | Meaning |
|---|---|
| required | Mandatory |
| placeholder | Hint text |
| readonly | Read only |
| disabled | Disabled field |
| maxlength | Max length |
| pattern | Regex rule |

### Advantages of HTML Forms

- Easy data collection
- No plugin required
- Client-side validation
- Backend integration
- User interaction

**Introduction to cascading style sheet**

**CSS -** CSS (Cascading Style Sheets) एक stylesheet language है जिसका उपयोग HTML द्वारा बनाए गए web page के presentation और visual appearance को नियंत्रित (control) करने के लिए किया जाता है। CSS के द्वारा web page को attractive, readable और user-friendly बनाया जाता है।

HTML का मुख्य कार्य web page का structure और content बनाना होता है, जबकि CSS का कार्य उस content को design, color, layout और formatting प्रदान करना होता है। इसी कारण CSS को HTML से अलग रखा गया है ताकि content और design को separate किया जा सके।

## CSS की आवश्यकता (Need / Importance of CSS)

- CSS का उपयोग web development में निम्नलिखित कारणों से आवश्यक है:
- CSS web page को professional look प्रदान करता है।
- CSS के द्वारा multiple web pages का design एक साथ control किया जा सकता है।
- CSS से code repetition कम होती है क्योंकि एक ही stylesheet को कई pages में apply किया जा सकता है।
- CSS website की maintenance और modification को आसान बनाता है।
- CSS website की loading speed बढ़ाता है क्योंकि styling code अलग file में होता है।
- CSS से responsive design संभव होता है जिससे website mobile, tablet और desktop पर सही दिखती है।

## CSS का कार्य सिद्धांत (Working Principle of CSS)

CSS HTML elements पर आधारित होकर कार्य करता है। CSS में किसी HTML element को selector के माध्यम से चुना जाता है और उस पर विभिन्न properties और values apply की जाती हैं।

## CSS Syntax:

```
selector {
    property: value;
}
```

जहाँ,

**Selector** – HTML element को select करता है

**Property** – जिस style को apply करना है

**Value** – property का मान

**Example:**

```
p {
   color: blue;
   font-size: 14px;
}
```

इस उदाहरण में सभी <p> tags का text blue color और 14px size में display होगा।

## CSS को HTML में शामिल करने के तरीके

CSS को HTML document में तीन तरीकों से शामिल किया जा सकता है:

### 1.Inline CSS

Inline CSS में CSS code को सीधे HTML element के अंदर style attribute के माध्यम से लिखा जाता है।

**Syntax:**

```
<h1 style="color:red; text-align:center;">Welcome</h1>
```

### विशेषताएँ:

- यह CSS केवल एक ही element पर लागू होती है
- Code repetition अधिक होती है
- Maintenance difficult होता है
- Inline CSS का प्रयोग सामान्यतः testing या small changes के लिए किया जाता है।

**2.Internal CSS**

Internal CSS में CSS code को HTML document के <head> section के अंदर <style> tag में लिखा जाता है।

**Syntax:**

```
<head>
  <style>
    p {
      color: green;
      font-size: 16px;
    }
  </style>
</head>
```

**विशेषताएँ:**

- यह CSS पूरे HTML page पर लागू होती है
- Multiple elements को एक साथ style किया जा सकता है
- Medium size websites के लिए उपयुक्त
- Internal CSS तब उपयोगी होती है जब एक ही page का अलग design चाहिए।

**3.External CSS**

External CSS में CSS code को एक अलग .css file में लिखा जाता है और HTML file से <link> tag के द्वारा जोड़ा जाता है।

**Syntax (HTML file):**

```
<head>
  <link rel="stylesheet" href="style.css">
</head>
```

**Syntax (style.css):**

```
body {
  background-color: lightgray;
}
```

```
h1 {
    color: navy;
}
```

**विशेषताएँ:**

- एक stylesheet को multiple web pages में use किया जा सकता है
- Code reusability और easy maintenance
- Professional और recommended method
- Large websites में External CSS सबसे अधिक उपयोग की जाती है

**Stylesheet -** Stylesheet एक ऐसी file होती है जिसमें HTML elements के लिए styling rules लिखे जाते हैं। जब यह file external होती है तो इसे External Stylesheet कहा जाता है।

Stylesheet का extension .css होता है और इसमें केवल CSS code लिखा जाता है।

**Stylesheet बनाने की प्रक्रिया**

- किसी text editor (Notepad, VS Code आदि) को open करें
- CSS rules लिखें
- File को .css extension के साथ save करें
  (उदाहरण: style.css)
- HTML file के <head> section में <link> tag के द्वारा stylesheet को attach करें

**CSS के लाभ (Advantages of CSS)**

- Website का design और layout बेहतर बनाता है
- Code duplication कम करता है
- Easy maintenance और updating
- Faster page loading
- Device independent presentation
- SEO friendly structure

**CSS Properties**

CSS Properties वे styling attributes होती हैं जिनका उपयोग HTML elements की visual appearance, layout तथा formatting को नियंत्रित करने के लिए किया जाता है। जब किसी HTML element पर CSS लागू की जाती है, तब वास्तविक styling कार्य CSS properties के माध्यम से ही किया जाता है।

CSS में प्रत्येक property यह निर्धारित करती है कि किसी element का कौन-सा aspect बदला जाएगा, जैसे text, spacing, border, position या layout से संबंधित विशेषताएँ। Properties हमेशा किसी value के साथ लिखी जाती हैं, जो यह बताती है कि वह property किस प्रकार लागू होगी।

**CSS Property का सामान्य स्वरूप (General Form)**

property : value;

यहाँ

- **Property** यह बताती है कि element का कौन-सा भाग style होगा
- **Value** उस property का मान निर्धारित करती है
- एक element पर एक से अधिक properties एक साथ लागू की जा सकती हैं और सभी properties को **semicolon (;)** से अलग किया जाता है।

**CSS Properties की भूमिका (Role of CSS Properties)**

- CSS properties का मुख्य उद्देश्य HTML document को presentation control प्रदान करना है। ये properties web page के निम्न पहलुओं को नियंत्रित करती हैं:
- Text और content की visual formatting
- Elements के बीच की spacing
- Page का layout और alignment
- Elements की positioning
- Website की consistency और uniform design

**CSS Background -** CSS Background Styling का उपयोग HTML elements की background appearance को control करने के लिए किया जाता है। इसके द्वारा किसी element के पीछे color, image, position, repeat और size जैसी properties apply की जाती हैं, जिससे web page अधिक attractive और readable बनता है।

Background styling को सामान्यतः body, div, section, table, form आदि elements पर apply किया जाता है।

**CSS Background Properties**

CSS में background को control करने के लिए विभिन्न properties उपलब्ध हैं, जिनका विवरण निम्नलिखित है:

**1.background-color**

यह property किसी element की background का रंग निर्धारित करती है।

**Syntax:**

```
selector {
    background-color: value;
}
```

**Example:**

```
body {
    background-color: lightblue;
}
```

**2.background-image**

इस property का उपयोग element की background में image set करने के लिए किया जाता है।

**Syntax:**

```
selector {
    background-image: url("image_path");
}
```

**Example:**

```
div {
    background-image: url("bg.jpg");
}
```

**3.background-repeat**

यह property यह निर्धारित करती है कि background image repeat होगी या नहीं।

**Syntax:**

```
selector {
    background-repeat: value;
}
```

**Example:**

```
div {
    background-repeat: no-repeat;
}
```

**4.background-position**

इस property के द्वारा background image की position तय की जाती है, जैसे left, right, center आदि।

**Syntax:**

```
selector {
    background-position: value;
}
```

**Example:**

```
div {
    background-position: center;
}
```

**5.background-size**

यह property background image का size निर्धारित करती है।

**Syntax:**

```
selector {
    background-size: value;
}
```

**Example:**

```
div {
    background-size: cover;
}
```

**CSS Text Formatting -** CSS Text Formatting का उपयोग HTML elements के text की appearance, alignment और presentation को नियंत्रित करने के लिए किया जाता है। इसके द्वारा text का color, alignment, spacing, decoration और transformation बदला जा सकता है, जिससे web page का content clear, readable और attractive बनता है।

Text formatting properties सामान्यतः paragraphs, headings, links और span elements पर लागू की जाती हैं।

**CSS Text Formatting Properties**

CSS में text को format करने के लिए विभिन्न properties उपलब्ध हैं, जिनका विवरण निम्नलिखित है:

**1.color**

यह property text का color निर्धारित करती है।

**Syntax:**

```
selector {
    color: value;
}
```

Example:

```
p {
    color: blue;
}
```

**2.text-align**

यह property text को left, right, center या justify में align करने के लिए उपयोग की जाती है।

**Syntax:**

```
selector {
    text-align: value;
}
```

**Example:**

```
p {
    text-align: justify;
}
```

**3.text-decoration**

इस property का उपयोग text पर underline, overline या line-through लगाने के लिए किया जाता है।

**Syntax:**

```
selector {
    text-decoration: value;
}
```

**Example:**

```
a {
    text-decoration: none;
}
```

**4.text-transform**

यह property text को uppercase, lowercase या capitalize में बदलने के लिए प्रयोग की जाती है।

**Syntax:**

```
selector {
    text-transform: value;
}
```

**Example:**

```
h1 {
    text-transform: uppercase;
}
```

**CSS Font Styling -** CSS Font Styling का उपयोग HTML elements के text की font appearance को नियंत्रित करने के लिए किया जाता है। इसके द्वारा text का font type, size, style, thickness और variant बदला जा सकता है, जिससे web page का content clear, readable और professional दिखाई देता है।

Font styling properties सामान्यतः headings, paragraphs, spans और links पर apply की जाती हैं।

**CSS Font Styling Properties**

CSS में font को control करने के लिए विभिन्न properties उपलब्ध हैं, जिनका विवरण निम्नलिखित है:

**1.font-family**

यह property text के लिए font type या font face निर्धारित करती है। इसमें multiple fonts दिए जा सकते हैं ताकि यदि पहला font उपलब्ध न हो तो दूसरा उपयोग हो सके।

**Syntax:**

```
selector {
    font-family: value;
}
```

**Example:**

```
body {
    font-family: Arial, Helvetica, sans-serif;
}
```

**2.font-size**

यह property text का size निर्धारित करती है।

**Syntax:**

```
selector {
    font-size: value;
}
```

**Example:**

```
p {
    font-size: 16px;
}
```

**3.font-style**

यह property text को normal, italic या oblique रूप में display करती है।

**Syntax:**

```
selector {
    font-style: value;
}
```

**Example:**

```
em {
    font-style: italic;
}
```

**4.font-weight**

यह property text की thickness या boldness निर्धारित करती है।

**Syntax:**

```
selector {
    font-weight: value;
}
```

**Example:**

```
h1 {
    font-weight: bold;
}
```

**CSS List Styling -** CSS List Styling का उपयोग HTML lists (<ul>, <ol>, <li>) की appearance और presentation को नियंत्रित करने के लिए किया जाता है। इसके द्वारा list के markers (bullets / numbers), position, style और custom images को बदला जा सकता है, जिससे list अधिक clear, attractive और readable बनती है।

List styling सामान्यतः navigation menus, content lists और itemized information में उपयोग की जाती है।

**CSS List Styling Properties**

CSS में lists को style करने के लिए विभिन्न properties उपलब्ध हैं, जिनका विवरण निम्नलिखित है:

**1.list-style-type**

यह property list के marker type को निर्धारित करती है, जैसे unordered list में bullet का type और ordered list में numbering का style।

**Syntax:**

```
selector {
    list-style-type: value;
}
```

**Example:**

```
ul {
    list-style-type: square;
}
```

**2.list-style-position**

यह property यह निर्धारित करती है कि list marker content के अंदर (inside) दिखाई देगा या content के बाहर (outside)।

**Syntax:**

```
selector {
    list-style-position: value;
}
```

**Example:**

```
ul {
    list-style-position: inside;
}
```

**3.list-style-image**

इस property का उपयोग default bullet या number की जगह custom image लगाने के लिए किया जाता है।

**Syntax:**

```
selector {
    list-style-image: url("image_path");
}
```

**Example:**

```
ul {
    list-style-image: url("bullet.png");
}
```

**CSS Table Styling** - CSS Table Styling का उपयोग HTML tables (<table>, <tr>, <th>, <td>) की appearance, layout और readability को नियंत्रित करने के लिए किया जाता है। इसके द्वारा table के borders, spacing, alignment, size और colors को नियंत्रित किया जा सकता है, जिससे table अधिक clear, structured और professional दिखाई देती है।

Table styling का प्रयोग सामान्यतः data presentation, reports और comparison tables में किया जाता है।

**CSS Table Styling Properties**

CSS में tables को style करने के लिए विभिन्न properties उपलब्ध हैं, जिनका विवरण निम्नलिखित है:

**1.border**

यह property table और उसके cells के चारों ओर border लगाने के लिए उपयोग की जाती है।

**Syntax:**

```
selector {
    border: value;
}
```

**Example:**

```
table, th, td {
    border: 1px solid black;
}
```

**2.border-collapse**

यह property यह निर्धारित करती है कि table के borders अलग-अलग (separate) दिखेंगे या merged (collapse) होकर एक border बनेंगे।

**Syntax:**

```
selector {
    border-collapse: value;
}
```

**Example:**

```
table {
    border-collapse: collapse;
}
```

**3.border-spacing**

यह property table cells के बीच की spacing को निर्धारित करती है। यह केवल तब कार्य करती है जब border-collapse: separate हो।

**Syntax:**

```
selector {
    border-spacing: value;
}
```

**Example:**

```
table {
    border-spacing: 10px;
}
```

**4.width और height**

इन properties का उपयोग table या cells का size निर्धारित करने के लिए किया जाता है।

**Syntax:**

```
selector {
    width: value;
    height: value;
}
```

**Example:**

```
table {
    width: 100%;
}
```

**5.text-align**

यह property table cell के अंदर text को left, right या center में align करने के लिए प्रयोग की जाती है।

**Syntax:**

```
selector {
    text-align: value;
}
```

**Example:**

```
th, td {
    text-align: center;
}
```

**CSS Selector -** CSS में Selector का उपयोग HTML elements को select करने के लिए किया जाता है ताकि उन पर styling apply की जा सके। ID और Class CSS के सबसे महत्वपूर्ण selectors हैं, जिनके द्वारा specific elements या element groups पर style लागू किया जाता है।

**1.CSS ID Selector**

ID एक unique identifier होता है, जिसका उपयोग किसी HTML document में एक ही element को uniquely पहचानने के लिए किया जाता है। एक HTML page में एक ID केवल एक बार ही उपयोग की जानी चाहिए।

ID selector का उपयोग तब किया जाता है जब किसी specific element पर special styling लागू करनी हो।

**Syntax (HTML):**

```
<tag id="idname">
```

**Syntax (CSS):**

```
#idname {
    property: value;
}
```

**Example:**

```
<p id="intro">This is introduction</p>

#intro {
   color: blue;
   font-size: 18px;
}
```

## CSS ID की विशेषताएँ

- ID unique होती है
- एक page में एक ही ID एक बार उपयोग की जाती है
- ID selector की priority (specificity) अधिक होती है
- Special या single element styling के लिए उपयोगी

### 2.CSS Class Selector

Class एक group selector होती है, जिसका उपयोग एक से अधिक HTML elements पर समान styling लागू करने के लिए किया जाता है। एक ही class name को multiple elements में reuse किया जा सकता है।

Class selector का उपयोग तब किया जाता है जब कई elements का same design या format हो।

**Syntax (HTML):**

```
<tag class="classname">
```

**Syntax (CSS):**

```
.classname {
   property: value;
}
```

**Example:**

```
<p class="content">Paragraph 1</p>

<p class="content">Paragraph 2</p>

.content {
```

```
    color: green;
    text-align: justify;
}
```

## CSS Class की विशेषताएँ

- Class reusable होती है
- Multiple elements पर apply की जा सकती है
- ID की तुलना में priority कम होती है
- Common styling के लिए उपयोगी

## ID और Class में अंतर (Difference)

| CSS ID | CSS Class |
|---|---|
| Unique होती है | Reusable होती है |
| एक element के लिए | Multiple elements के लिए |
| # symbol से define | . symbol से define |
| Higher specificity | Lower specificity |
| Special styling के लिए | Common styling के लिए |

**CSS Box Model -** CSS Box Model एक ऐसा concept है जो यह बताता है कि browser किसी भी HTML element को एक rectangular box के रूप में कैसे render करता है। Web page पर दिखाई देने वाला हर element—जैसे paragraph, div, image या heading—एक box के अंदर होता है।

इस box का size, spacing और positioning CSS Box Model के नियमों के अनुसार तय होता है। इसलिए Box Model को web layout designing की foundation माना जाता है।

## CSS Box Model के मुख्य भाग

CSS Box Model चार मुख्य layers से मिलकर बना होता है (अंदर से बाहर की ओर):

- Content
- Padding
- Border
- Margin

**1.Content Area**

Content area box model का सबसे अंदर का भाग होता है, जिसमें actual text, image या data दिखाई देता है।
Content का size सामान्यतः width और height properties से नियंत्रित किया जाता है।

**Example:**

```
div {
   width: 300px;
   height: 150px;
}
```

यह केवल content का size निर्धारित करता है, इसमें padding, border और margin शामिल नहीं होते।

**2.Padding (Inner Space)**

Padding content और border के बीच की अंदरूनी जगह (inner spacing) होती है। Padding element के अंदर होती है और background color padding तक apply होता है।

**Padding Properties**

- padding-top
- padding-right
- padding-bottom
- padding-left

**Example 1: Individual Padding**

```
div {
   padding-top: 10px;
   padding-right: 20px;
   padding-bottom: 10px;
   padding-left: 20px;
}
```

**Example 2: Shorthand Padding**

```
div {
    padding: 15px;
}
```

**Example 3: Different values**

```
div {
    padding: 10px 20px;
}
```

यहाँ

10px → top & bottom

20px → left & right

**3.Border (Boundary)**

Border padding और margin के बीच स्थित होती है। यह element के चारों ओर एक visible boundary बनाती है, जिससे element अलग-से पहचाना जा सकता है।

**Border Properties**

- border-width
- border-style
- border-color

**Example 1: Separate Border Properties**

```
div {
    border-width: 2px;
    border-style: solid;
    border-color: black;
}
```

**Example 2: Shorthand Border**

```
div {
    border: 3px dashed red;
}
```

**Example 3: Different borders on sides**

```
div {
    border-top: 2px solid blue;
    border-bottom: 2px solid green;
}
```

**4.Margin (Outer Space)**

Margin element के बाहर की बाहरी जगह (outer spacing) होती है, जो एक element को दूसरे element से दूरी प्रदान करती है। Margin में background color apply नहीं होता।

**Margin Properties**

- margin-top
- margin-right
- margin-bottom
- margin-left

**Example 1: Individual Margin**

```
div {
    margin-top: 20px;
    margin-bottom: 20px;
}
```

**Example 2: Shorthand Margin**

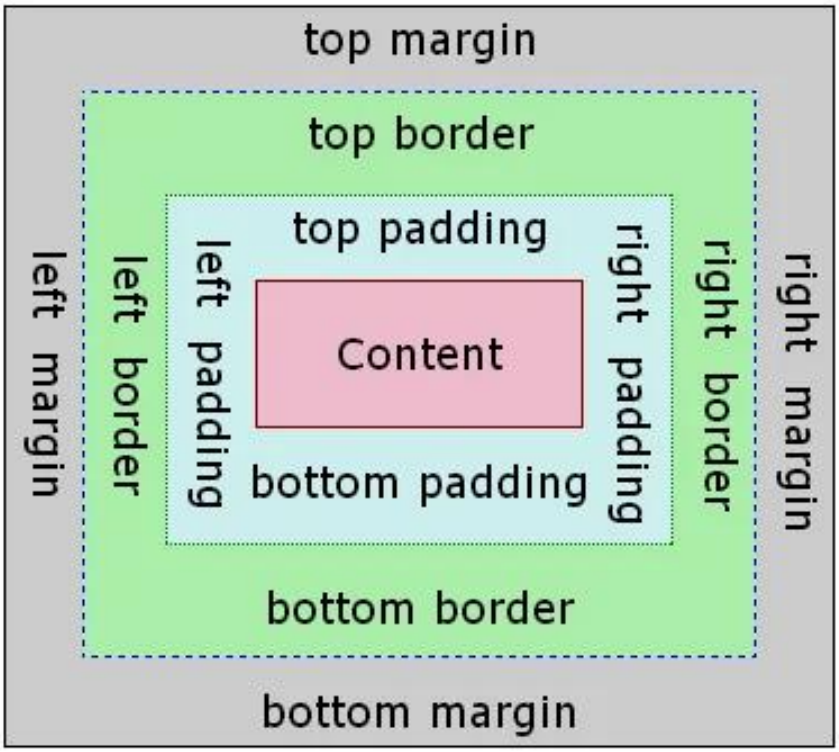
```
div {
    margin: 15px;
}
```

**Example 3: Center Alignment using Margin**

```
div {
    width: 300px;
    margin: auto;
}
```

यह element को horizontally center कर देता है।

**Box Model का Visual Structure**



**CSS Box Model का महत्व**

- Web page layout को समझने में मदद करता है
- Element spacing और alignment control करता है
- Responsive design का आधार
- Clean और professional UI बनाने में सहायक

**CSS Grouping -** CSS Grouping का अर्थ है एक से अधिक HTML elements या selectors को एक साथ group करके उन पर समान CSS properties apply करना। इसमें same styling को बार-बार अलग-अलग selectors के लिए लिखने के बजाय उन्हें comma ( , ) से अलग करके एक ही rule में लिखा जाता है।

CSS Grouping का मुख्य उद्देश्य code repetition को कम करना, stylesheet को concise बनाना तथा maintenance को आसान बनाना है।

**CSS Grouping की आवश्यकता (Need of CSS Grouping)**

- CSS Grouping की आवश्यकता निम्न कारणों से होती है:
- जब कई HTML elements का design और formatting समान हो
- Styling code की duplication को avoid करने के लिए
- Stylesheet को short, readable और organized बनाने के लिए
- Future modification को easy बनाने के लिए

**CSS Grouping का Syntax**

```
selector1, selector2, selector3 {
    property: value;
}
```

यहाँ सभी selectors पर एक ही CSS rules लागू होते हैं।

**Example 1: Multiple Elements Grouping**

```
h1, h2, h3 {
    color: blue;
    text-align: center;
}
```

इस उदाहरण में h1, h2 और h3 सभी पर समान color और alignment लागू होगा।

**Example 2: Grouping with Class and Element**

```
p, .content {
    font-size: 16px;
    line-height: 1.5;
}
```

यहाँ paragraph और class content वाले सभी elements पर same styling लागू होगी।

**Example 3: Grouping with ID and Class**

```
#header, .menu {
    background-color: lightgray;
}
```

ID और class दोनों को group किया जा सकता है।

**CSS Grouping बनाम Repetition (Without vs With Grouping)**

**Without Grouping**

```
h1 {
    color: red;
}
h2 {
    color: red;
}
h3 {
    color: red;
}
```

**With Grouping**

```
h1, h2, h3 {
    color: red;
}
```

Grouping से code short और efficient बनता है।

**CSS Grouping के लाभ (Advantages)**

- Code repetition कम होती है
- Stylesheet compact और readable बनती है
- Maintenance और updating आसान होती है
- Consistent design सुनिश्चित होता है
- Professional coding practice को promote करता है

**CSS Dimension -** CSS Dimension का उपयोग HTML elements के size (आकार) को नियंत्रित करने के लिए किया जाता है। इसके द्वारा यह तय किया जाता है कि कोई element web page पर कितनी चौड़ाई (width) और ऊँचाई (height) में दिखाई देगा।

CSS dimensions layout designing का एक महत्वपूर्ण भाग हैं, क्योंकि element का size ही उसके positioning, alignment और responsiveness को प्रभावित करता है।

**CSS Dimension Properties**

CSS में element के size को नियंत्रित करने के लिए मुख्यतः निम्न properties उपयोग की जाती हैं:

- width
- height
- min-width
- max-width
- min-height
- max-height

**Width Property**

width property element की content area की चौड़ाई निर्धारित करती है। इसमें padding, border और margin शामिल नहीं होते (default box model में)।

**Syntax:**

```
selector {
    width: value;
}
```

**Example:**

```
div {
    width: 300px;
}
```

**Height Property**

height property element की content area की ऊँचाई निर्धारित करती है।

**Syntax:**

```
selector {
    height: value;
}
```

**Example:**

```
div {
    height: 150px;
}
```

**Minimum Width (min-width)**

min-width यह निर्धारित करती है कि element की width किसी निश्चित मान से कम नहीं होगी, चाहे screen size छोटा हो जाए।

**Syntax:**

```
selector {
    min-width: value;
}
```

**Example:**

```
div {
    min-width: 200px;
}
```

**Maximum Width (max-width)**

max-width यह तय करती है कि element की width किसी निश्चित सीमा से अधिक नहीं होगी, चाहे screen size बड़ा क्यों न हो।

**Syntax:**

```
selector {
    max-width: value;
}
```

**Example:**

```
div {
    max-width: 800px;
}
```

**Minimum Height (min-height)**

min-height यह सुनिश्चित करती है कि element की height कम से कम दिए गए मान के बराबर होगी।

**Syntax:**

```
selector {
    min-height: value;
}
```

**Example:**

```
div {
    min-height: 100px;
}
```

**Maximum Height (max-height)**

max-height यह निर्धारित करती है कि element की height एक निश्चित सीमा से अधिक नहीं होगी।

**Syntax:**

```
selector {
   max-height: value;
}
```

**Example:**

```
div {
   max-height: 300px;
}
```

**CSS Dimension में Units**

CSS dimension values विभिन्न units में दी जा सकती हैं, जैसे:

- **px (pixels)** – fixed size
- **% (percentage)** – parent element के अनुसार
- **em, rem** – font size पर आधारित
- **vw, vh** – viewport के अनुसार

**Example:**

```
div {
   width: 50%;
   height: 50vh;
}
```

**CSS Dimension का महत्व (Importance)**

- Element size को accurately control करता है
- Page layout को structured बनाता है
- Responsive design में सहायक
- Overflow और alignment issues को manage करता है
- Professional UI design में आवश्यक

**CSS Display -** CSS Display Property यह निर्धारित करती है कि कोई HTML element web page पर कैसे दिखाई देगा और कैसे behave करेगा। Display property के द्वारा यह तय होता है कि element:

- नई line से शुरू होगा या नहीं
- पूरी width लेगा या केवल content जितनी
- एक ही line में अन्य elements के साथ रहेगा या अलग
- दिखेगा या पूरी तरह hide होगा

Display property web page के layout और structure को नियंत्रित करने में महत्वपूर्ण भूमिका निभाती है।

**Display Property की आवश्यकता (Need of Display)**

- CSS display का उपयोग निम्न कारणों से आवश्यक है:
- Page layout को control करने के लिए
- Inline और block elements के behavior को बदलने के लिए
- Navigation menus और layouts बनाने के लिए
- Elements को show / hide करने के लिए
- Responsive design implement करने के लिए

**Common Display Values**

CSS display property के प्रमुख values निम्नलिखित हैं:

- block
- inline
- inline-block
- none

**display: block**

जब किसी element को display: block दिया जाता है, तब वह element, हमेशा new line से शुरू होता है**,** पूरी available width लेता है|

Width और height दोनों apply हो सकती हैं

**Syntax:**

```
selector {
    display: block;
}
```

**Example:**

```
span {
    display: block;
}
```

यहाँ span (जो normally inline होता है) block की तरह behave करेगा।

**display: inline**

display: inline में element नई line से start नहीं करता केवल content जितनी width लेता है | Width और height apply नहीं होती

**Syntax:**

```
selector {
    display: inline;
}
```

**Example:**

```
div {
    display: inline;
}
```

यहाँ div inline element की तरह behave करेगा।

**display: inline-block**

inline-block में inline और block दोनों के features होते हैं, एक ही line में रहता है

Width और height apply की जा सकती हैं

**Syntax:**

```
selector {
    display: inline-block;
}
```

**Example:**

```
div {
    display: inline-block;
    width: 150px;
    height: 100px;
}
```

Navigation menu और cards बनाने में बहुत उपयोगी।

**display: none**

display: none element को पूरी तरह hide कर देता है। Element Screen पर दिखाई नहीं देता

Page layout में कोई space नहीं लेता

**Syntax:**

```
selector {
    display: none;
}
```

**Example**

```
p {
    display: none;
}
```

**CSS Display का महत्व (Importance)**

- Page layout control करता है
- Inline और block behavior बदलता है
- Responsive design में सहायक
- Clean और flexible UI बनाने में मदद करता है

**CSS Positioning -** CSS Positioning वह mechanism है जिसके द्वारा हम यह निर्धारित करते हैं कि कोई HTML element web page पर कहाँ और कैसे place होगा। Positioning के माध्यम से elements को normal document flow से हटाकर या उसी flow में रखते हुए specific स्थान (top, bottom, left, right) पर रखा जा सकता है।

CSS positioning layout designing, menus, pop-ups, banners, tooltips आदि बनाने में अत्यंत महत्वपूर्ण है।

### CSS Positioning की आवश्यकता (Need)

CSS positioning की आवश्यकता निम्न कारणों से होती है:

- Elements को exact location पर रखने के लिए
- Floating menus, headers, footers बनाने के लिए
- Images और text को overlap कराने के लिए
- Fixed navigation bar बनाने के लिए
- Dynamic और responsive layout design करने के लिए

### Position Property का Syntax

```
selector {
    position: value;
    top: value;
    bottom: value;
    left: value;
    right: value;
}
```

top, bottom, left, right position property के साथ ही काम करते हैं।

### CSS Position के प्रकार (Types of Position)

CSS में मुख्यतः पाँच प्रकार की positioning होती है:

- static
- relative
- absolute
- fixed
- sticky

**position: static**

यह CSS की default position value है। Static positioned elements Normal document flow में रहते हैं|

top, left, right, bottom properties apply नहीं होती |

**Syntax:**

```
selector {
    position: static;
}
```

**Example:**

```
p {
    position: static;
}
```

इसमें element अपने default स्थान पर ही रहता है।

**position: relative**

Relative positioning में element Normal document flow में रहता है | अपने original position के relative move करता है| Original space बना रहता है |

**Syntax:**

```
selector {
    position: relative;
}
```

**Example:**

```
div {
    position: relative;
    top: 20px;
    left: 30px;
}
```

Element अपने original स्थान से 20px नीचे और 30px दाईं ओर move होगा।

**position: absolute**

Absolute positioning में element Normal document flow से बाहर हो जाता है | Nearest positioned ancestor के relative होता है| अगर ancestor positioned नहीं है, तो body के relative होता है |

**Syntax:**

```
selector {
    position: absolute;
}
```

**Example:**

```
.container {
    position: relative;
}

.box {
    position: absolute;
    top: 10px;
    right: 20px;
}
```

.box अपने parent .container के relative position होगा।

**position: fixed**

Fixed positioning में element, Browser window के relative होता है | Scroll करने पर भी अपनी जगह नहीं बदलता | Normal document flow से बाहर होता है |

**Syntax:**

```
selector {
    position: fixed;
}
```

**Example:**

```
nav {
    position: fixed;
    top: 0;
    width: 100%;
}
```

Fixed navigation bar बनाने में उपयोगी।

**position: sticky**

Sticky positioning, Relative और fixed का combination है | Scroll करते समय एक point तक relative रहता है | उस point के बाद fixed की तरह behave करता है |

**Syntax:**

```
selector {
    position: sticky;
}
```

**Example**

```
h1 {
    position: sticky;
    top: 0;
}
```

Page scroll करने पर heading top पर stick हो जाएगी।

**CSS Positioning का महत्व (Importance)**

- Page layout को control करता है
- Overlapping designs बनाने में सहायक
- Fixed headers और footers के लिए उपयोगी
- Modern UI design का आधार

**CSS Float -** CSS Float Property का उपयोग किसी HTML element को left या right side में move करने के लिए किया जाता है, जिससे अन्य content (जैसे text या elements) उसके around wrap हो सके।

Float property का मुख्य उद्देश्य text wrapping और column-based layouts बनाना है। CSS Flexbox और Grid से पहले, float का उपयोग layout design के लिए बहुत अधिक किया जाता था।

### CSS Float की आवश्यकता (Need of Float)

CSS float की आवश्यकता निम्नलिखित कारणों से होती है:

- Image के चारों ओर text wrap कराने के लिए
- Two-column या multi-column layout बनाने के लिए
- Navigation menu बनाने के लिए
- Sidebars और content area align करने के लिए
- Elements को left या right side align करने के लिए

### Float Property का Syntax

```
selector {
    float: value;
}
```

**Possible Values:**

- left
- right
- none
- inherit

### float: left

जब किसी element को float: left दिया जाता है, तो वह element:

- अपने container के left side में चला जाता है
- उसके बाद आने वाला content उसके right side में wrap हो जाता है

**Syntax:**

```
selector {
    float: left;
}
```

**Example:**

```
img {
   float: left;
   margin-right: 15px;
}
```

Image left में जाएगी और text उसके right में wrap होगा।

**float: right**

float: right में element:

- Container के right side में चला जाता है
- Content उसके left side में wrap होता है

**Syntax:**

```
selector {
   float: right;
}
```

**Example:**

```
div {
   float: right;
   width: 200px;
}
```

Sidebar right side में align हो जाएगा।

**float: none**

यह default value होती है। Element:

- Normal document flow में रहता है
- Floating effect remove हो जाता है

**Syntax:**

```
selector {
    float: none;
}
```

**Float का महत्व (Importance)**

- Text wrapping के लिए अत्यंत उपयोगी
- Simple layouts बनाने में सहायक
- Image alignment में useful
- Legacy layouts को समझने के लिए आवश्यक

**CSS Alignment -** CSS Alignment का अर्थ है web page पर elements या text को सही दिशा और सही position में व्यवस्थित करना। Alignment के द्वारा हम यह नियंत्रित करते हैं कि content: left, right, center या justify में दिखे, vertically top, middle या bottom में align हो container के अंदर कैसे arrange हो |

CSS alignment का उपयोग text alignment, image alignment, div alignment और layout design में किया जाता है।

**CSS Alignment की आवश्यकता (Need)**

CSS alignment की आवश्यकता निम्न कारणों से होती है:

- Web page को readable और attractive बनाने के लिए
- Text और images को proper position में रखने के लिए
- Navigation menu और buttons align करने के लिए
- Layout को professional look देने के लिए
- Responsive design implement करने के लिए

**CSS में Alignment के प्रकार**

CSS में alignment को मुख्य रूप से निम्न भागों में समझा जा सकता है:

- Text Alignment
- Horizontal Alignment (Elements)
- Vertical Alignment

**Text Alignment (text-align)**

text-align property का उपयोग text और inline elements को horizontally align करने के लिए किया जाता है।

**Possible Values:**

- left
- right
- center
- justify

**Syntax:**

```
selector {
    text-align: value;
}
```

**Example:**

```
p {
    text-align: justify;
}
```

Text दोनों किनारों से evenly align हो जाता है।

**Horizontal Alignment of Elements**

**(A) Using margin -** Block level elements को horizontally center करने के लिए

**Syntax:**

```
selector {
    margin: auto;
}
```

**Example:**

```
div {
    width: 300px;
    margin: auto;
}
```

Div horizontally center में आ जाता है।

**(B) Using float**

**Example**

```
.box {
    float: right;
}
```

Element right side align हो जाता है।

**Vertical Alignment (vertical-align)**

vertical-align property का उपयोग inline या table-cell elements को vertically align करने के लिए किया जाता है।

**Common Values:**

- top
- middle
- bottom

**Syntax:**

```
selector {
    vertical-align: value;
}
```

**Example:**

```
img {
    vertical-align: middle;
}
```

Image text के middle में align हो जाती है।

**Alignment in Table Cells**

Table के अंदर content align करने के लिए:

**Example**

```
td {
   text-align: center;
   vertical-align: middle;
}
```

Table cell का content horizontally और vertically center में होगा।

**CSS Alignment का महत्व (Importance)**

- Layout को clean और structured बनाता है
- UI/UX बेहतर करता है
- Content readability बढ़ाता है
- Responsive designs में सहायक

**CSS Pseudo-Class**

CSS Pseudo-Class का उपयोग किसी HTML element की special state या condition को define करने के लिए किया जाता है। यह element के actual content या structure को बदले बिना, उसके behavior या appearance को बदलने में मदद करता है।

**Pseudo-class यह बताती है कि element:**

- किसी विशेष स्थिति (state) में है
- User interaction के अनुसार बदल रहा है
- Document structure के अनुसार select हुआ है

**Pseudo-Class की आवश्यकता**

- User interaction (hover, click, focus) handle करने के लिए
- Links की अलग-अलग states को style करने के लिए
- Form elements की active या focus state दिखाने के लिए
- Structure-based styling (first element, last element आदि) के लिए
- Dynamic और interactive UI बनाने के लिए

**Pseudo-Class का Syntax**

```
selector:pseudo-class {
    property: value;
}
```

Pseudo-class हमेशा colon ( : ) से शुरू होती है।

**Link-Related Pseudo-Classes**

**(A) :link -** Unvisited link को style करने के लिए।

```
a:link {
    color: blue;
}
```

**(B) :visited -** Visited link को style करता है।

```
a:visited {
    color: purple;
}
```

**(C) :hover -** जब mouse element पर ले जाया जाता है।

```
a:hover {
    color: red;
}
```

**(D) :active -** जब link को click किया जाता है।

```
a:active {
    color: green;
}
```

Link pseudo-classes को हमेशा इस order में लिखना चाहिए: LVHA (Link, Visited, Hover, Active)

User Interaction Pseudo-Classes

**(A) :focus -** जब input field या element focus में हो।

```
input:focus {
    background-color: lightyellow;
}
```

**(B) :checked -** Checked radio button या checkbox को style करता है।

```
input:checked {
    outline: 2px solid green;
}
```

**(C) :disabled -** Disabled form element को select करता है।

```
input:disabled {
    background-color: lightgray;
}
```

**Structural Pseudo-Classes**

**(A) :first-child -** Parent का पहला child element select करता है।

```
p:first-child {
    color: red;
}
```

**(B) :last-child -** Parent का आखिरी child element select करता है।

```
p:last-child {
    color: blue;
}
```

**(C) :nth-child(n) -** Specific position वाले elements को select करता है।

```
li:nth-child(2) {
    color: green;
}
```

**Form-Related Pseudo-Classes**

**(A) :required**

```
input:required {
    border: 2px solid red;
}
```

**(B) :valid और :invalid**

```
input:valid {
    border: 2px solid green;
}

input:invalid {
    border: 2px solid red;
}
```

**Pseudo-Class का महत्व (Importance)**

- Interactive UI बनाने में सहायक
- JavaScript के बिना dynamic effects
- Forms को user-friendly बनाता है
- Code को clean और efficient बनाता है

## CSS Navigation Bar

CSS Navigation Bar (Navbar) एक ऐसा component है जिसका उपयोग website के different pages या sections के links को व्यवस्थित (organize) करने के लिए किया जाता है। Navigation bar आमतौर पर web page के top, left, right या bottom में स्थित होता है।

Navigation bar website की usability और user experience को बेहतर बनाता है, क्योंकि इसके माध्यम से user आसानी से एक page से दूसरे page पर जा सकता है।

## Navigation Bar की आवश्यकता

- Website pages के बीच आसान navigation के लिए
- Website structure को clear दिखाने के लिए
- Professional और consistent layout बनाने के लिए
- User experience (UX) सुधारने के लिए
- Responsive web design implement करने के लिए

## Navigation Bar बनाने की Basic Structure

Navigation bar सामान्यतः unordered list (<ul>) और list items (<li>) का उपयोग करके बनाई जाती है।

HTML Structure

```
<ul class="navbar">
  <li><a href="#">Home</a></li>
  <li><a href="#">About</a></li>
  <li><a href="#">Services</a></li>
  <li><a href="#">Contact</a></li>
</ul>
```

## CSS द्वारा Navigation Bar Styling

### Basic CSS

```
.navbar {
  list-style-type: none;
  margin: 0;
```

```
    padding: 0;
}
```

list-style-type: none bullets हटाता है।

**Horizontal Navigation Bar**

Horizontal navigation bar में menu items एक ही line में left से right arrange होते हैं। इसके लिए display या float का उपयोग किया जाता है।

```
.navbar li {
    float: left;
}

.navbar li a {
    display: block;
    padding: 10px 20px;
    text-decoration: none;
}
```

सभी menu items horizontally align हो जाते हैं।

**Vertical Navigation Bar**

Vertical navigation bar में links column format में display होते हैं, जो अक्सर sidebar के रूप में उपयोग किया जाता है।

```
.navbar {
    width: 200px;
}

.navbar li a {
    display: block;
```

```
    padding: 10px;
}
```

**Fixed Navigation Bar**

Fixed navbar scroll करने पर भी same position पर रहती है।

```
.navbar {
    position: fixed;
    top: 0;
    width: 100%;
}
```

**Navigation Bar का महत्व (Importance)**

- Website navigation आसान बनाता है
- User interaction improve करता है
- Website को professional look देता है
- Content organization में सहायक

**CSS Image Sprites**

CSS Image Sprite एक ऐसी technique है जिसमें multiple small images को एक single large image में combine कर दिया जाता है और फिर CSS की मदद से उस large image के अलग-अलग हिस्सों को web page पर display किया जाता है।

इस technique का मुख्य उद्देश्य HTTP requests को कम करना और website की loading speed को तेज़ करना होता है।

**Image Sprites की आवश्यकता**

- Web page की loading speed बढ़ाने के लिए
- Multiple image requests को reduce करने के लिए
- Server load कम करने के लिए
- Performance optimization के लिए
- Icons और buttons efficiently display करने के लिए

**Image Sprite कैसे काम करता है?**

Image sprite में:

- सभी images एक single file में होती हैं
- background-image के रूप में sprite image set की जाती है
- background-position से image का specific हिस्सा दिखाया जाता है

**Image Sprite का Syntax**

```
selector {
   background-image: url("sprite.png");
   background-repeat: no-repeat;
   background-position: x y;
}
```

**Image Sprite का Example**

**Step 1: Sprite Image**

मान लीजिए एक image icons.png है जिसमें:

Home icon (0px, 0px)

Search icon (-50px, 0px)

Mail icon (-100px, 0px)

**Step 2: CSS Code**

```
.icon {
   background-image: url("icons.png");
   background-repeat: no-repeat;
   width: 50px;
   height: 50px;
}
```

```
.home {
    background-position: 0px 0px;
}

.search {
    background-position: -50px 0px;
}

.mail {
    background-position: -100px 0px;
}
```

**Step 3: HTML Code**

```
<div class="icon home"></div>
<div class="icon search"></div>
<div class="icon mail"></div>
```

अलग-अलग icons single image से display होंगे।

**Image Sprites के Advantages**

- HTTP requests कम होती हैं
- Website loading fast होती है
- Server performance बेहतर होती है
- Icons और buttons consistent रहते हैं
- Bandwidth consumption कम होता है

**Image Sprites के Disadvantages**

- Sprite image बनाना complex हो सकता है
- Image update करने में पूरी sprite बदलनी पड़ सकती है
- Large sprite file manage करना कठिन होता है

**Image Sprites का उपयोग कहाँ किया जाता है?**

- Navigation bar icons
- Social media icons
- Buttons और hover effects
- Toolbars और menus

**CSS Attribute Selector**

CSS Attribute Selector का उपयोग HTML elements को उनके attributes या attribute values के आधार पर select करने के लिए किया जाता है। इसका अर्थ है कि हम element के tag name या class के बजाय, उसके attribute (जैसे type, href, title, value आदि) के आधार पर styling apply कर सकते हैं।

Attribute selectors CSS को more flexible और powerful बनाते हैं, क्योंकि यह बिना extra class या id जोड़े elements को target करने की सुविधा देते हैं।

**Attribute Selector की आवश्यकता**

- Specific attributes वाले elements को style करने के लिए
- बिना class/id लगाए selective styling करने के लिए
- Forms elements को efficiently design करने के लिए
- Link types के आधार पर styling करने के लिए
- Clean और reusable CSS लिखने के लिए

**Attribute Selector का Basic Syntax**

```
selector[attribute] {
    property: value;
}
```

या

```
selector[attribute="value"] {
    property: value;
}
```

Square brackets [ ] attribute selector को define करते हैं।

**Types of CSS Attribute Selectors**

CSS में attribute selectors के मुख्यतः 7 प्रकार होते हैं:

```
[attribute]

[attribute="value"]

[attribute~="value"]

[attribute|="value"]

[attribute^="value"]

[attribute$="value"]

[attribute*="value"]
```

**[attribute] Selector**

यह selector उन elements को select करता है जिनमें specified attribute मौजूद होता है, चाहे उसका value कुछ भी हो।

**Syntax**

```
selector[attribute] {
    property: value;
}
```

**Example**

```
input[required] {
    border: 2px solid red;
}
```

सभी required inputs को style करेगा।

**[attribute="value"] Selector**

यह selector उन elements को select करता है जिनका attribute exactly specified value के बराबर होता है।

**Syntax**

```
selector[attribute="value"] {
    property: value;
}
```

**Example**

```
input[type="text"] {
    background-color: lightyellow;
}
```

**[attribute~="value"] Selector**

यह selector तब match करता है जब attribute value space-separated list में specified word मौजूद हो।

**Syntax**

```
selector[attribute~="value"] {
    property: value;
}
```

**Example**

```
div[class~="box"] {
    border: 1px solid black;
}
```

Matches: class="box big"
Not matches: class="textbox"

**[attribute|="value"] Selector**

यह selector attribute value को match करता है जो:

- exactly value हो
- या value से शुरू होकर - (hyphen) के साथ हो
- Mostly language attributes के लिए उपयोग होता है।

**Syntax**

```
selector[attribute|="value"] {
    property: value;
}
```

**Example**

```
p[lang|="en"] {
    color: blue;
}
```

Matches: lang="en" , lang="en-US"

**[attribute^="value"] Selector**

यह selector उन elements को select करता है जिनका attribute value given value से start होता है।

**Syntax**

```
selector[attribute^="value"] {
    property: value;
}
```

**Example**

```
a[href^="https"] {
    color: green;
}
```

Secure links को highlight करने के लिए उपयोगी।

**[attribute$="value"] Selector**

यह selector attribute value को match करता है जो given value पर end होता है।

**Syntax**

```
selector[attribute$="value"] {
    property: value;
}
```

**Example**

```
a[href$=".pdf"] {
    color: red;
}
```

PDF links को style करेगा।

**[attribute*="value"] Selector**

यह selector attribute value में कहीं भी substring मौजूद हो तो match करता है।

**Syntax**

```
selector[attribute*="value"] {
    property: value;
}
```

**Example**

```
img[src*="icon"] {
    border: 2px solid blue;
}
```

जिन images के नाम में "icon" है, वे select होंगे।

**Attribute Selector का महत्व (Importance)**

- HTML को बिना modify किए styling संभव
- Form elements styling आसान
- Clean और maintainable CSS
- Advanced selection capability

**CSS Color**

CSS Color का उपयोग web page के विभिन्न elements जैसे text, background, border, outline, shadow आदि का रंग निर्धारित (define) करने के लिए किया जाता है। Color property website के look, feel और readability को सीधे प्रभावित करती है।

CSS में colors का प्रयोग करके web page को attractive, meaningful और user-friendly बनाया जाता है।

**CSS Color की आवश्यकता (Need)**

- Text को readable बनाने के लिए
- Background और foreground में contrast देने के लिए
- Important elements को highlight करने के लिए
- User experience बेहतर बनाने के लिए
- Visual hierarchy बनाने के लिए

**CSS में Color Apply करने के तरीके**

- Color Names
- RGB Colors
- HEX Colors
- HSL Colors
- RGBA Colors
- HSLA Colors

**Color Name Method**

CSS में कुछ predefined color names होते हैं जैसे: red, blue, green, black आदि।

**Syntax**

```
selector {
    color: red;
}
```

**Example**

```
p {
    color: blue;
}
```

Simple और readable तरीका, लेकिन limited colors।

**RGB Color Method**

RGB का अर्थ है Red, Green, Blue। इसमें हर color का value 0 से 255 के बीच होता है।

**Syntax**

```
selector {
    color: rgb(red, green, blue);
}
```

**Example**

```
h1 {
    color: rgb(255, 0, 0);
}
```

Red color को दर्शाता है।

**HEX Color Method**

HEX color hexadecimal format में लिखा जाता है, जो # से शुरू होता है।
Format: #RRGGBB

**Syntax**

```
selector {
    color: #RRGGBB;
}
```

**Example**

```
p {
    color: #00FF00;
}
```

Green color दर्शाता है।

**HSL Color Method**

HSL का अर्थ है:

- Hue (color type: 0–360)
- Saturation (color intensity %)
- Lightness (brightness %)

**Syntax**

```
selector {
    color: hsl(hue, saturation, lightness);
}
```

**Example**

```
h2 {
    color: hsl(240, 100%, 50%);
}
```

Blue color दर्शाता है।

**RGBA Color Method**

RGBA, RGB का extended version है जिसमें Alpha (opacity) भी शामिल होता है।

Alpha value: 0.0 (transparent) से 1.0 (opaque)

**Syntax**

```
selector {
    color: rgba(r, g, b, a);
}
```

**Example**

```
div {
    background-color: rgba(255, 0, 0, 0.5);
}
```

Semi-transparent red background।

**HSLA Color Method**

HSLA, HSL का extended version है जिसमें Alpha (opacity) शामिल होती है।

**Syntax**

```
selector {
    color: hsla(hue, saturation, lightness, alpha);
}
```

**Example**

```
p {
    color: hsla(120, 100%, 25%, 0.8);
}
```

**CSS Color का महत्व (Importance)**

- Web page को visually appealing बनाता है
- User attention guide करता है
- Branding और identity दिखाता है
- Readability और usability बढ़ाता है

Creating Page Layout and Site Designs

Page Layout से तात्पर्य web page के विभिन्न elements जैसे header, navigation bar, content area, sidebar और footer की व्यवस्थित arrangement से है। जबकि Site Design पूरे website के overall structure, look, feel और user interaction pattern को दर्शाता है।

Page layout यह तय करता है कि content कहाँ दिखाई देगा, और site design यह सुनिश्चित करता है कि पूरा website consistent, attractive और user-friendly हो।

**Page Layout और Site Design की आवश्यकता**

- Creating page layout और site design की आवश्यकता निम्न कारणों से होती है:
- Website content को clear और structured रूप में दिखाने के लिए
- User को आसानी से information ढूँढने में सहायता देने के लिए
- Website को professional और visually appealing बनाने के लिए
- Responsive और scalable design implement करने के लिए
- Better user experience (UX) और usability प्राप्त करने के लिए

**Page Layout के मुख्य Components**

एक standard web page layout में सामान्यतः निम्न sections होते हैं:

**1. Header**

Website का top हिस्सा

Logo, title, search bar आदि होते हैं

**2. Navigation Bar**

Website pages के links

Horizontal या vertical हो सकता है

**3. Main Content Area**

Website का मुख्य भाग

Text, images, videos, articles आदि

**4. Sidebar**

Additional information, links या ads

Left या right side में होता है

**5. Footer**

Website का bottom section

Copyright, contact info, policies आदि

**CSS का Role in Page Layout Design**

- Elements की positioning तय करता है
- Width, height और spacing control करता है
- Content alignment manage करता है
- Responsive behavior enable करता है

**Site Design के Important Principles**

**1. Consistency**

Same colors, fonts और layout पूरे site में

**2. Simplicity**

Clean और clutter-free design

**3. Readability**

Proper font size और color contrast

**4. Navigation**

Easy और clear navigation structure

**5. Responsiveness**

Mobile, tablet और desktop सभी पर proper view

**Page Layout और Site Design का महत्व (Importance)**

- Website usability बढ़ाता है
- User engagement improve करता है
- Content hierarchy clear करता है
- Professional web presence बनाता है

# Introduction to Web Publishing or Hosting

## Creating the Web Site

Creating the Web Site का अर्थ है एक ऐसी प्रक्रिया जिसमें planning, designing और development के माध्यम से एक complete website तैयार की जाती है, जिसे बाद में internet पर publish किया जा सके।इस प्रक्रिया में website के purpose, content, structure, design और functionality को ध्यान में रखा जाता है।

Web site creation केवल web pages बनाने तक सीमित नहीं है, बल्कि यह एक systematic process है जिसमें user requirements और technical aspects दोनों शामिल होते हैं।

## Web Site Creation की आवश्यकता

- Information को worldwide publish करने के लिए
- Educational, business या personal purpose के लिए
- Online presence establish करने के लिए
- Communication और promotion के लिए
- Digital content को structured तरीके से प्रस्तुत करने के लिए

## Web Site Creation के मुख्य Steps

Web site creation एक step-by-step process होती है:

### Step 1: Planning of Web Site

इस step में website का purpose और goals तय किए जाते हैं।

Planning में निम्न बातें शामिल होती हैं:

- Website का उद्देश्य (educational, business, personal)
- Target audience
- Website का type (static या dynamic)
- Required pages (Home, About, Contact आदि)

Proper planning से website effective और user-friendly बनती है।

**Step 2: Designing the Web Site**

Designing phase में website का layout और appearance तय किया जाता है।

इसमें शामिल हैं:

- Page layout design
- Color scheme selection
- Fonts और text style
- Navigation menu design

Designing का मुख्य उद्देश्य website को attractive और readable बनाना होता है।

**Step 3: Creating Web Pages**

इस step में actual web pages बनाए जाते हैं।

- HTML का उपयोग structure बनाने के लिए
- CSS का उपयोग styling और layout के लिए
- Images, text, links आदि add किए जाते हैं

प्रत्येक web page website के overall design का हिस्सा होता है।

**Step 4: Organizing Content**

Content organization का अर्थ है information को logical और systematic order में रखना।

इसमें:

- Headings और sub-headings का प्रयोग
- Paragraphs और lists का सही उपयोग
- Related information को एक साथ रखना

Proper content organization website की readability बढ़ाता है।

**Step 5: Linking Web Pages**

इस step में web pages को आपस में hyperlinks के माध्यम से जोड़ा जाता है।

- Internal links (same website pages)
- External links (other websites)

Linking से website navigation आसान हो जाती है।

**Tools Used for Creating Web Site**

Web site creation के लिए विभिन्न tools का उपयोग किया जाता है:

- Text Editors (HTML, CSS लिखने के लिए)
- Web designing software
- Image editing tools
- Browsers (testing के लिए)

**Saving the Site**

Saving the Site का अर्थ है web site के सभी components (web pages, images, CSS files, scripts आदि) को proper folder structure के साथ computer या server पर सुरक्षित रूप से store करना।
यह web site creation का एक महत्वपूर्ण step है, क्योंकि बिना सही तरीके से save किए गए files website को सही ढंग से run या publish नहीं किया जा सकता।

**Site को Save करने की आवश्यकता**

- Website data को loss से बचाने के लिए
- Web pages को browser में सही से open करने के लिए
- Hosting server पर upload करने के लिए
- Website को future में update करने के लिए
- सभी files के बीच linking को maintain करने के लिए

Proper Folder Structure for Saving Site

एक standard web site folder structure इस प्रकार होती है:

Root Folder (Website Name)

index.html

about.html

contact.html

css/

images/

js/

Proper folder structure से website manage करना आसान होता है।

**Advantages of Proper Site Saving**

- Easy website management
- Faster uploading on server
- Error-free navigation
- Easy future modification
- Secure data storage

**Creating Web Site Structure**

Web Site Structure का अर्थ है website के सभी web pages, files और folders को एक सुव्यवस्थित (organized) ढंग से arrange करना, ताकि website को आसानी से navigate, manage और maintain किया जा सके।

Creating web site structure website development का एक planning-oriented step है, जो यह तय करता है कि:

- कौन-कौन से pages होंगे
- वे एक-दूसरे से कैसे जुड़े होंगे
- files कहाँ store होंगी

**Web Site Structure का महत्व**

- User को website समझने और navigate करने में आसानी होती है
- Website professional और well-organized दिखती है
- Content management आसान हो जाता है
- Website future expansion के लिए ready रहती है
- SEO (Search Engine Optimization) बेहतर होता है

**Types of Web Site Structure**

**Hierarchical Structure**

यह सबसे common structure है, जिसमें pages को parent-child relationship में arrange किया जाता है।

**उदाहरण:**

- Home Page
- About Us
- Services
- Contact

Large websites के लिए उपयुक्त।

**Linear Structure**

इस structure में pages को एक sequence में arrange किया जाता है।

**उदाहरण:**

Page 1 → Page 2 → Page 3

Tutorials और step-by-step websites के लिए उपयोगी।

**Network (Webbed) Structure**

इस structure में हर page एक-दूसरे से link होता है।

Blogs और content-heavy websites में उपयोग किया जाता है।

**Common Web Site Structure Example**

एक सामान्य website structure इस प्रकार होता है:

Website Root

index.html

about.html

contact.html

services.html

css/

images/

js/

यह structure small और medium websites के लिए ideal है।

**Best Practices for Creating Web Site Structure**

- Simple और logical structure बनाएँ
- Limited menu items रखें
- Consistent navigation रखें
- Meaningful page names रखें
- Future expansion को ध्यान में रखें

**Creating Titles for Web Pages**

Web Page Title वह text होता है जो web page के browser tab, title bar और search engine results में दिखाई देता है।
HTML में page title को <title> tag के द्वारा define किया जाता है, जो <head> section के अंदर लिखा जाता है।

Page title web page की identity और purpose को दर्शाता है।

**Web Page Title का महत्व**

- Browser tab में page को identify करता है
- Search engines को page का topic बताता है
- User को content का idea देता है
- Website की professionalism बढ़ाता है
- Bookmark करते समय यही नाम save होता है

**HTML में Title Tag**

HTML में हर web page का title <title> tag से बनाया जाता है।

**Syntax**

```
<head>
  <title>Page Title</title>
</head>
```

<title> tag हमेशा <head> section के अंदर ही होना चाहिए।

**Example of Web Page Title**

```
<html>
<head>
  <title>Computer Fundamentals - Home</title>
</head>
<body>
  Web page content
</body>
</html>
```

यहाँ “Computer Fundamentals - Home” page का title है।

**Characteristics of a Good Web Page Title**

एक अच्छा web page title निम्न गुणों वाला होना चाहिए:

- Short और meaningful
- Page content को clearly represent करे
- Unique हो (हर page का अलग title)
- Keywords include करे
- User-friendly हो

**Types of Web Page Titles**

**Main Page (Home Page) Title**

Home page का title website के नाम को दर्शाता है।

**उदाहरण:**

ABC College | Official Website

**Section Page Title**

Section pages website के specific भाग को दर्शाते हैं।

**उदाहरण:**

About Us | ABC College

**Content-Based Title**

यह title content के विषय पर आधारित होता है।

**उदाहरण:**

Introduction to HTML | Web Design Notes

**Role of Titles in SEO (Search Engine Optimization)**

- Search engines title को सबसे पहले पढ़ते हैं
- Keywords title में होने से ranking improve होती है
- Proper title से click-through rate बढ़ता है

इसलिए title concise और keyword-rich होना चाहिए।

**Publishing a Website**

Publishing a Website का अर्थ है तैयार की गई web site को local computer से web server पर upload करना, ताकि वह internet पर live हो जाए और कोई भी user उसे browser के माध्यम से access कर सके।

जब website successfully publish हो जाती है, तब उसे Live Website कहा जाता है।

**Publishing a Website की आवश्यकता**

Website publish करना आवश्यक है क्योंकि:

- Website को public access प्रदान करना
- Information को global level पर share करना
- Business / institution की online presence बनाना
- 24×7 services उपलब्ध कराना
- Digital communication को बढ़ावा देना

**Requirements for Publishing a Website**

Website publish करने से पहले निम्न चीज़ें आवश्यक होती हैं:

**1. Completed Web Site**

HTML pages

CSS files

Images और scripts

**2. Domain Name**

Website का unique address

जैसे: www.collegename.com

**3. Web Hosting Service**

Server space जहाँ website files store होती हैं

**4. Internet Connection**

Files upload करने के लिए आवश्यक

## Types of Web Hosting

### Shared Hosting

- Multiple websites एक ही server पर
- Low cost
- Small websites के लिए suitable

### Dedicated Hosting

- पूरा server एक website के लिए
- High performance
- Large organizations के लिए

### Cloud Hosting

- Multiple servers का use
- High reliability
- Scalable hosting

## Steps in Publishing a Website

### Step 1: Selecting Web Hosting Provider

Hosting provider server space और services प्रदान करता है।

**उदाहरण:**

Storage space

Bandwidth

FTP access

### Step 2: Registering Domain Name

Domain name website की पहचान होता है।

**उदाहरण:**

www.mywebsite.com

**Step 3: Preparing Website Files**

Publish करने से पहले:

- All files properly saved हों
- Links और images check हों
- Home page index.html हो

**Step 4: Uploading Website Files**

Website files को server पर upload किया जाता है:

FTP (File Transfer Protocol) software

Hosting control panel (File Manager)

Files आमतौर पर public_html folder में upload की जाती हैं।

**Step 5: Connecting Domain to Hosting**

Domain name को hosting server से connect किया जाता है।

DNS settings configure की जाती हैं

Propagation में कुछ समय लग सकता है

**Step 6: Testing the Live Website**

Publish होने के बाद:

Website open करके check की जाती है

All pages और links test किए जाते हैं

**Publishing Tools and Methods**

Website publish करने के लिए विभिन्न tools का उपयोग किया जाता है:

- FTP clients
- Hosting dashboards
- Web authoring tools

Tools publishing को आसान बनाते हैं।

**Advantages of Publishing a Website**

- Global reach
- Easy information sharing
- Business promotion
- Cost-effective communication
- Professional image