HTML & CSS Study Notes

HTML (HyperText Markup Language)

HTML Document Structure

- `<!DOCTYPE html>` : Declares HTML5 document type
- `<html>` : Root element containing all content
- `<head>` : Contains metadata (not visible on page)
- `<body>` : Contains visible page content

```html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Page Title</title>
</head>
<body>
<!-- Content goes here -->
</body>
</html>
```

Meta Tags

Meta tags provide metadata about the HTML document. They are placed in the `<head>` section and are not displayed on the page but are used by browsers, search engines, and other web services.

Common Meta Tag Attributes:

Character Encoding:

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

- Specifies the character encoding for the HTML document
- UTF-8 is the most widely used encoding that supports all characters and symbols
- Should be placed early in the `<head>` section

Viewport (for Responsive Design):

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

- Controls how the page is displayed on mobile devices
- `width=device-width` : Sets the width to follow the screen width of the device
- `initial-scale=1.0` : Sets the initial zoom level when the page is first loaded

Description:

```html
<meta name="description" content="Learn HTML basics with examples.">
```

- Provides a brief description of the page content
- Used by search engines in search results snippets
- Should be 150-160 characters for optimal display in search results
- Important for SEO (Search Engine Optimization)

Keywords:

```html
<meta name="keywords" content="HTML, CSS, Web Development">
```

- Specifies keywords relevant to the page content
- Separated by commas
- Less important for modern SEO but still used by some search engines
- Should be relevant to actual page content

Author:

```html
<meta name="author" content="Gautam Mukherjee">
```

- Specifies the author of the document
- Useful for content attribution and contact information
- Can be a person's name, organization, or email address

Other Useful Meta Tags:

```html
```

```
<!-- Refresh page every 30 seconds -->
<meta http-equiv="refresh" content="30">

<!-- Redirect to another page after 5 seconds -->
<meta http-equiv="refresh" content="5;url=https://example.com">

<!-- Control caching -->
<meta http-equiv="cache-control" content="no-cache">

<!-- Open Graph for social media -->
<meta property="og:title" content="Page Title">
<meta property="og:description" content="Page description">
<meta property="og:image" content="image.jpg">
```

Headings

Six levels: `<h1>` (largest) to `<h6>` (smallest)

- Use hierarchically for proper structure
- Important for SEO and accessibility

```html
<h1>Main Heading</h1>
<h2>Sub Heading</h2>
<h3>Sub-sub Heading</h3>
<!-- h4, h5, h6 for smaller headings -->
```

Paragraphs

```html
<p>This is a paragraph of text.</p>
<p>This is another paragraph.</p>
```

- `<p>` creates paragraph blocks
- Automatically adds spacing between paragraphs
- Can contain inline elements like `<strong>`, `<em>`

Links

```html
```

```html
<a href="https://example.com">External Link</a>
<a href="page.html">Internal Link</a>
<a href="#section">Anchor Link</a>
<a href="mailto:email@example.com">Email Link</a>
```

- `href` attribute specifies destination
- `target="_blank"` opens in new tab
- Use descriptive link text for accessibility

Lists

Unordered Lists:

```html
<ul>
<li>Item 1</li>
<li>Item 2</li>
<li>Item 3</li>
</ul>
```

Ordered Lists:

```html
<ol>
<li>First item</li>
<li>Second item</li>
<li>Third item</li>
</ol>
```

Images

```html
<img src="image.jpg" alt="Description of image" width="300" height="200">
```

- `src` : Image file path
- `alt` : Alternative text for accessibility
- `width` / `height` : Optional size attributes
- Always include alt text

Tables

```html
<table>
<thead>
<tr>
<th>Header 1</th>
<th>Header 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data 1</td>
<td>Data 2</td>
</tr>
</tbody>
</table>
```

- `<table>` : Container for table
- `<tr>` : Table row
- `<th>` : Table header cell
- `<td>` : Table data cell

Forms

```html
```

```
<form action="/submit" method="POST">
<!-- Text Input -->
<input type="text" name="username" placeholder="Enter username">

<!-- Email Input -->
<input type="email" name="email" required>

<!-- Password Input -->
<input type="password" name="password">

<!-- Number Input -->
<input type="number" name="age" min="0" max="120">

<!-- Radio Buttons -->
<input type="radio" name="gender" value="male" id="male">
<label for="male">Male</label>
<input type="radio" name="gender" value="female" id="female">
<label for="female">Female</label>

<!-- Checkboxes -->
<input type="checkbox" name="subscribe" id="subscribe">
<label for="subscribe">Subscribe to newsletter</label>

<!-- Textarea -->
<textarea name="message" rows="4" cols="50" placeholder="Your message"></textarea>

<!-- Select Dropdown -->
<select name="country">
<option value="us">United States</option>
<option value="ca">Canada</option>
<option value="uk">United Kingdom</option>
</select>

<!-- Submit Button -->
<button type="submit">Submit</button>
<button type="reset">Reset</button>
</form>
```

Video

```
html
```

```html
<video width="320" height="240" controls>
<source src="movie.mp4" type="video/mp4">
<source src="movie.ogg" type="video/ogg">
Your browser does not support the video tag.
</video>
```

- `controls` : Shows play/pause controls
- Multiple `<source>` elements for browser compatibility

Audio

```html
<audio controls>
<source src="audio.mp3" type="audio/mpeg">
<source src="audio.ogg" type="audio/ogg">
Your browser does not support the audio element.
</audio>
```

- Similar to video but for audio files
- `autoplay`, `loop` attributes available

Marquee Tag

The `<marquee>` tag creates scrolling text or images. **Note: This tag is deprecated in HTML5 and should be avoided in modern web development.** Use CSS animations instead.

```html
<!-- Basic marquee -->
<marquee>This text will scroll from right to left</marquee>

<!-- Marquee with attributes -->
<marquee direction="up" behavior="scroll" scrollamount="3">
Scrolling upward text
</marquee>

<!-- Marquee with styling -->
<marquee direction="left" bgcolor="yellow" width="50%" height="30">
Styled scrolling text
</marquee>
```

Marquee Attributes:

- `direction` : left (default), right, up, down

- `behavior` : scroll (default), slide, alternate
- `scrollamount` : Speed of scrolling (1-10, default is 6)
- `scrolldelay` : Delay between each scroll movement in milliseconds
- `loop` : Number of times to scroll (-1 for infinite, default)
- `bgcolor` : Background color
- `width` : Width of the marquee area
- `height` : Height of the marquee area

Modern Alternative (CSS Animation):

```css
.scroll-text {
  animation: scroll-left 10s linear infinite;
}

@keyframes scroll-left {
  0% { transform: translateX(100%); }
  100% { transform: translateX(-100%); }
}
```

CSS (Cascading Style Sheets)

What is CSS

CSS is a styling language used to control the presentation and layout of HTML documents. It separates content (HTML) from presentation (CSS), making websites more maintainable and flexible.

CSS Implementation Methods

Inline CSS:

```html
<p style="color: red; font-size: 16px;">Styled text</p>
```

Internal CSS:

```html
```

```
<head>
<style>
p { color: blue; font-size: 18px; }
</style>
</head>
```

External CSS:

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

- External is most preferred for maintainability
- Inline has highest priority, external has lowest

CSS Syntax and Selectors

```css
/* Basic syntax */
selector {
property: value;
property: value;
}

/* Element selector */
p { color: red; }

/* Class selector */
.my-class { font-size: 16px; }

/* ID selector */
#my-id { background: yellow; }

/* Descendant selector */
div p { margin: 10px; }

/* Multiple selectors */
h1, h2, h3 { font-family: Arial; }
```

Colors, Background, and Fonts

Colors:

```css
.text {
color: red; /* Named color */
color: #ff0000; /* Hex color */
color: rgb(255,0,0); /* RGB color */
color: rgba(255,0,0,0.5); /* RGBA with transparency */
}
```

Background:

```css
.container {
background-color: #f0f0f0;
background-image: url('image.jpg');
background-repeat: no-repeat;
background-position: center;
background-size: cover;
}
```

Fonts:

```css
.text {
font-family: Arial, sans-serif;
font-size: 16px;
font-weight: bold;
font-style: italic;
text-align: center;
text-decoration: underline;
}
```

CSS Box Model and Layout

Box Model Components:

```css
```

```
.box {
width: 200px;
height: 100px;
padding: 20px; /* Space inside the border */
border: 2px solid black;
margin: 10px; /* Space outside the border */
}
```

Display Properties:

```css
.block { display: block; } /* Full width, new line */
.inline { display: inline; } /* Only content width, same line */
.inline-block { display: inline-block; } /* Hybrid of both */
```

Positioning:

```css
.relative { position: relative; top: 10px; left: 20px; }
.absolute { position: absolute; top: 0; right: 0; }
.fixed { position: fixed; bottom: 0; left: 0; }
.sticky { position: sticky; top: 0; }
```

Hover Effects

```css
.button {
background-color: blue;
transition: background-color 0.3s ease;
}

.button:hover {
background-color: darkblue;
cursor: pointer;
}

.link:hover {
color: red;
text-decoration: underline;
}
```

Flexbox

```css
.flex-container {
display: flex;
justify-content: center; /* Horizontal alignment */
align-items: center; /* Vertical alignment */
flex-direction: row; /* Direction of flex items */
flex-wrap: wrap; /* Allow wrapping */
gap: 10px; /* Space between items */
}

.flex-item {
flex: 1; /* Grow to fill space */
flex-basis: 200px; /* Initial size */
}
```

Common Flexbox Values:

- `justify-content` : flex-start, center, flex-end, space-between, space-around
- `align-items` : flex-start, center, flex-end, stretch
- `flex-direction` : row, column, row-reverse, column-reverse

Google Fonts

Steps to use Google Fonts:

1. Visit fonts.google.com
2. Select desired fonts and weights
3. Copy the link tag to HTML head
4. Use font-family in CSS

```html
<!-- In HTML head -->
<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;700&display=swap" rel="stylesheet">
```

```css
```

```
/* In CSS */
body {
font-family: 'Roboto', sans-serif;
}

.light { font-weight: 300; }
.normal { font-weight: 400; }
.bold { font-weight: 700; }
```

Quick Reference Tips

- Always use semantic HTML elements
- Include alt text for images
- Use external CSS for better maintainability
- Test responsive design on different screen sizes
- Validate HTML and CSS code regularly
- Use developer tools for debugging
- Follow naming conventions for classes and IDs

Author: Gautam Mukherjee