

CHAPTER 2: BUILDING BLOCKS OF THE WEB

Introduction Overview of Web Development:

Web development involves creating and maintaining websites and web applications. It is divided into frontend development (clientside) and backend development (serverside). The front end includes everything users experience directly, such as layout, design, and interactivity. The back end consists of databases, servers, and application logic that power the front end.

Understanding the core technologies of HTML, CSS, and JavaScript is essential for any web developer. These technologies form the backbone of web development, enabling developers to create structured content, style it, and add interactivity.

Section 1: Creating Lists and Tables

Creating Lists in HTML

Lists in HTML are a way to group related items. There are three types of lists:

1. **Ordered Lists ():** Used for items that need to be presented in a specific sequence.
2. **Unordered Lists ():** Used for items that do not need to be in a specific order.
3. **Definition Lists (<dl>):** Used for pairs of terms and descriptions, commonly for glossaries or terms.

Ordered Lists ()

Ordered lists are used to create numbered lists. Each item is wrapped in an (list item) element.

Tags and Their Uses:

- : Defines the start and end of the ordered list.
- : Defines an item within the list.

Example:

Html

Copy

```
<ol>
  <li>Wake up</li>
  <li>Brush teeth</li>
  <li>Have breakfast</li>
</ol>
```

In this example:

- `` starts the ordered list.
- Each `` tag defines an item in the list.
- The browser automatically numbers the list items.

Unordered Lists ()


Unordered lists are used to create bulleted lists. Each item is wrapped in an `` (list item) element.

Tags and Their Uses:

- ``: Defines the start and end of the unordered list.
- ``: Defines an item within the list.

Example:

Html

 Copy

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

In this example:

- `` starts the unordered list.
- Each `` tag defines an item in the list.
- The browser automatically adds bullets to the list items.

Definition Lists (<dl>)

Definition lists are used to create a list of terms and their definitions. Terms are wrapped in `<dt>` (definition term) elements, and descriptions are wrapped in `<dd>` (definition description) elements.

Tags and Their Uses:

- `<dl>`: Defines the start and end of the definition list.
- `<dt>`: Defines a term in the list.
- `<dd>`: Defines the description of the term.

```
Html Copy

<dl>
  <dt>HTML</dt>
  <dd>HyperText Markup Language</dd>
  <dt>CSS</dt>
  <dd>Cascading Style Sheets</dd>
  <dt>JavaScript</dt>
  <dd>A programming language for web development</dd>
</dl>
```

In this example:

- `<dl>` starts the definition list.
- `<dt>` tags define the terms.
- `<dd>` tags define the descriptions for each term.

Creating Tables in HTML

Tables in HTML are used to display tabular data. Each table is defined with a `<table>` tag, and rows are defined with `<tr>` (table row) tags. Table headers are defined with `<th>` (table header) tags, and table data is defined with `<td>` (table data) tags.

Tags and Their Uses:

- `<table>`: Defines the start and end of the table.
- `<tr>`: Defines a row in the table.
- `<th>`: Defines a header cell in the table.
- `<td>`: Defines a data cell in the table.
- `<thead>`: Groups the header content.
- `<tbody>`: Groups the body content.
- `<tfoot>`: Groups the footer content

Example:

```
Html Copy

<table>
  <thead>
    <tr>
      <th>Name</th>
      <th>Age</th>
      <th>City</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>John</td>
      <td>30</td>
      <td>New York</td>
    </tr>
    <tr>
      <td>Jane</td>
      <td>25</td>
      <td>Los Angeles</td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td colspan="3">Data Footer</td>
    </tr>
  </tfoot>
</table>
```

In this example:

- `<table>` starts the table.
- `<thead>` groups the header row defined by `<tr>` and header cells defined by `<th>`.
- `<tbody>` groups the body rows defined by `<tr>` and data cells defined by `<td>`.
- `<tfoot>` groups the footer row defined by `<tr>` and a cell that spans three columns (using `colspan`).

Explanation of the Tags:

- `<table>`: This tag encloses the entire table.
- `<thead>`: This tag contains the rows that make up the table header. It helps separate the header from the body and footer for better readability and styling.
- `<tbody>`: This tag contains the rows that make up the main body of the table.

- `<tfoot>`: This tag contains the rows that make up the table footer. It's useful for summarizing the data in the table.
- `<tr>`: Each row in the table is defined using this tag.
- `<th>`: This tag defines a header cell. Header cells are typically bold and centered.
- `<td>`: This tag defines a standard data cell in the table.
- `colspan`: An attribute of `<td>` or `<th>` that allows a cell to span multiple columns.

Section 2: CSS (Cascading Style Sheets)

Introduction to CSS: CSS (Cascading Style Sheets) is the language used to describe the presentation of a web page, including colors, layout, and fonts. It allows you to create visually engaging web pages by separating the content (HTML) from the presentation (CSS).

CSS Syntax and Selectors

CSS Syntax: CSS syntax consists of selectors, properties, and values.

- **Selectors:** Indicate which HTML element(s) the style should apply to.
- **Properties:** Define the aspect of the element to be styled.
- **Values:** Specify the settings for the properties.

- **Selectors:** Indicate which HTML element(s) the style should apply to.
- **Properties:** Define the aspect of the element to be styled.
- **Values:** Specify the settings for the properties.

Example:

```
Css Copy
h1 {
  color: blue;
  font-size: 20px;
}
```

In this example:

- `h1` is the selector, targeting all `<h1>` elements.
- `color` and `font-size` are properties.
- `blue` and `20px` are values for the respective properties.

Types of Selectors

- **Element Selectors:** Apply styles to all instances of a specific HTML element.

Types of Selectors

Element Selectors: Apply styles to all instances of a specific HTML element.

Example:

Css

Copy

```
p {  
  color: green;  
}
```

- **Class Selectors:** Apply styles to all elements with a specific class attribute. Class names start with a dot (.).

Example:

Css

Copy

```
.text-highlight {  
  background-color: yellow;  
}
```

This targets all elements with the class `text-highlight` and sets their background color to yellow.

HTML:

Html

Copy

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

- **ID Selectors:** Apply styles to a single element with a specific ID attribute. IDs start with a hash (#)

```
#main-header {  
  text-align: center;  
  color: red;  
}
```

This targets the element with the ID `main-header` and centers its text and sets its color to red.

HTML:

Html

Copy

```
<h1 id="main-header">Welcome to My Website</h1>
```

- **Attribute Selectors:** Apply styles to elements with a specific attribute.

Attribute Selectors: Apply styles to elements with a specific attribute.

Example:

Css

Copy

```
input[type="text"] {  
  border: 1px solid black;  
}
```

This targets all `<input>` elements with the attribute `type="text"` and sets their border to 1px solid black.

HTML:

Html

Copy

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

Styling Text

- **Font Properties:**

Styling Text

Font Properties:

- `font-family` : Specifies the typeface.

Example:

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

- `font-size` : Defines the size of the text.

Example:

```
Css Copy
p {
  font-size: 14px;
}
```

➤ Text Properties:

- `text-align` : Aligns text horizontally (left, center, right).

Example:

```
Css Copy
h2 {
  text-align: center;
}
```

- `text-decoration` : Adds decoration to text (e.g., underline).

Example:

```
Css Copy
a {
  text-decoration: underline;
}
```

Box Model

The box model describes the structure of elements in terms of boxes. It consists of:

Components:

- **Content:** The actual content of the box.
- **Padding:** Space between the content and the border.
- **Border:** Surrounds the padding and content.
- **Margin:** Space outside the border.

Example:

```
Css Copy  
  
div {  
  padding: 10px;  
  border: 2px solid black;  
  margin: 20px;  
}
```

Understanding Box Sizing: The `box-sizing` property controls how the box model is calculated.

Example:

```
Css Copy  
  
div {  
  box-sizing: border-box;  
}
```

Setting `box-sizing: border-box` ensures that the padding and border are included in the element's total width and height.

CSS GRID

is a powerful layout system designed to create two dimensional grid based layouts.

Grid Container: Defined using ``display: grid`` or ``display: inline grid``.

Grid Items: Direct children of the grid container that are placed into the grid structure.

Grid Properties:

- **Grid Template Columns/Rows:** Defines the number and size of columns/rows in the grid.

Example: ``gridtemplatecolumns: 100px 200px;``

- **Gap:** Defines the spacing between grid items.

Example: ``gap: 10px;``

- **Grid Areas:** Named areas within the grid layout.

Example: ``gridtemplateareas: "header header" "sidebar content";``

Placing Items:

- **Grid Column/Row Start/End:** Specifies the start and end positions of grid items.

Example: ``gridcolumnstart: 1; gridcolumnend: 3;``

Other Useful Properties:

- **Justify Content:** Aligns the grid within its container horizontally.

Example: ``justifycontent: center;``

- **Align Content:** Aligns the grid within its container vertically.

Example: ``aligncontent: center;``

Sure! Here's an expanded overview of JavaScript concepts with examples included, but not in script format:

SECTION 3: INTRODUCTION TO JAVASCRIPT

- **JavaScript:** A versatile programming language used to create dynamic and interactive web pages.
- **Role of JavaScript:** Enhances the behavior of web pages by manipulating the DOM, validating forms, and handling user events.

JavaScript Syntax and Basics

- **Variables:** Containers for storing data values. For example, you can store the number 10 in a variable called `x`.
Keywords include `var`, `let`, and `const`.
- **Data Types:** Different kinds of data that can be stored in variables. These include strings (e.g., "Hello"), numbers (e.g., 5), booleans (e.g., true), undefined, and null.
- **Operators:** Symbols used to perform operations on variables and values. For instance, the `+` operator can be used to add two numbers.

Control Structures

- **Conditional Statements:** Used to perform different actions based on conditions. An example is using an `if` statement to check if a number is greater than 10 and printing a message if it is.
Types include `if`, `else if`, `else`, and `switch`.
- **Looping Statements:** Used to repeat a block of code. You might use a `for` loop to print numbers 1 to 5.
Types include `for`, `while`, and `do...while`.

Functions

- **Defining Functions:** Blocks of code designed to perform a particular task. For example, a function called `greet` can take a name and return a greeting message.
- **Function Declaration:** Defined using the keyword `function`.
- **Function Expression:** Defined as a variable holding a function.

- **Function Parameters and Arguments:** Values passed to functions. If a function ``add`` takes two parameters, you can pass the numbers 5 and 3 as arguments.
- **Return Values:** The output of a function. A function that calculates the sum of two numbers returns the result.

DOM Manipulation

- **Document Object Model (DOM):** A representation of the webpage that JavaScript can interact with.
- **Selecting Elements:** Methods to access HTML elements. For example, you can select an element with the ID ``myElement`` using ``getElementById``. Methods include ``getElementById()``, ``querySelector()``, and ``getElementsByName()``.
- **Modifying Content and Attributes:** Changing the content and attributes of elements. For instance, you can change the text content of an element to "New content" or set its class attribute to "newClass".
- **Event Handling:** Responding to user actions, such as clicking a button. You can attach an event listener to a button that prints a message when clicked. Methods include ``addEventListener()``.

Additional Concepts

- **Arrays:** Collections of items stored in a single variable. An array might contain the numbers 1, 2, and 3.
- **Objects:** Collections of keyvalue pairs. An object representing a person might have keys ``name``, ``age``, and ``city`` with corresponding values.
- **ES6 Features:** Modern JavaScript features.

- **Template Literals:** Used for string interpolation, such as embedding a variable within a string using backticks.
- **Arrow Functions:** A concise way to write functions using the `=>` syntax.
- **Destructuring:** Extracting values from arrays or objects into distinct variables.