UNIT 8

Web Development with HTML, CSS, and JavaScript

Student Learning Outcomes

By the end of this chapter, you will be able to:

- Understand JavaScript syntax and data types.
- Work with variables, operators, and functions in JavaScript.
- Handle events and user input with JavaScript.
- Create simple programs using JavaScript.
- Create HTML forms and style them.
- Use JavaScript to handle events with operators, variables, and functions.
- Develop static web pages.
- Apply HTML tags appropriately to create web pages.
- Create a basic HTML page.
- Add text, images, and links to a page.
- Create lists and tables.
- Apply styles to HTML elements.
- Work with fonts, colors, and backgrounds.
- Create web pages to display data in the paragraphs and lists.
- Familiarize students with CSS syntax.
- Create layouts with CSS.
- Add animations and transitions with CSS.
- Develop, test, and debug static web pages.
- Organize images and text effectively.
- Use JavaScript along with HTML to handle events using operators, variables, and functions.

Introduction

In this chapter, the fundamentals of web development, Hyper Text Markup Language (HTML), JavaScript and Cascading Style Sheet (CSS) will be explained. By the end of this chapter, you will be able to understand JavaScript syntax and data types, work with variables, operators, and functions, handle events and user inputs, create simple programs, and develop static web pages using HTML and CSS.

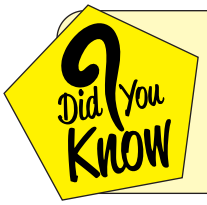
8.1 Web Development

Process of creating websites and web applications is called Web development. It means using various programming languages and tools to design, build, and maintain websites.

8.1.1 Why Learn Web Development?

Web development is a valuable skill for several reasons:

- **Digital Literacy:** When you learn web development, you find out how websites are made. You learn about HTML, which is like the skeleton of a web page, CSS, which makes the web page look nice, and JavaScript, which makes the web page interactive. This helps you understand how the internet works.
- **Career Opportunities:** Opens up a wide range of job prospects in the growing IT industry. Web developers can get many different kinds of jobs. You can become a web developer, web designer, and more. Many companies need web developers to create and maintain their websites. This means you can find good jobs in many places.
- **Problem-Solving:** When you build a website, you solve many problems. For example, if a website is slow, you figure out why and fix it. This helps you think logically and solve problems better.
- **Creativity:** Allows you to create visually appealing and interactive websites. Web development lets you be creative. You can design websites with cool layouts, colors, and interactive features. For example, you can create a personal blog or a portfolio to show your artwork, making your own unique website.
- **Entrepreneurship:** With web development skills, you can start your own online business. For example, if you make crafts, you can build a website to sell them. Or, you can create a new web service, like a fun app, and share it with the world.



Hotmail, one of the first web-based email services, was created by students Sabeer Bhatia and Jack Smith while they were at Stanford University. It was later acquired by Microsoft for $400 million.

8.2 Basic Components of Web Development

Web development involves creating websites and web applications. It has three main components:

1. **Front-end Development:** This focuses on what users see and interact with on a website. The following fundamentals are used to design interactive Front-ends:
   - **HTML** structures the content on web pages, like headings, paragraphs, images, and links.
   - **CSS** styles the content on web pages, changing colors, fonts, and layout to enhance the appearance.
   - **JavaScript** adds interactivity to web pages, making them dynamic and engaging. It allows features such as forms, animations, and games.
2. **Back-end Development:** This manages the behind-the-scenes functinality of a website, including servers, databases, and application logic. Key back-end technologies are:
   - **Web Servers** are computers that store and deliver web pages to users when they enter a URL.
   - **Databases** store and manage data, like user information, product details, and website content.
   - **Back-end** Programming Languages like PHP, Python, and Ruby handle tasks such as processing forms, and managing user logins.

Did You Know

The first website was created by Tim Berners-Lee in 1991 and it is still accessible at http: //info.cern.ch. It was a simple page with links to information about the World Wide Web project.

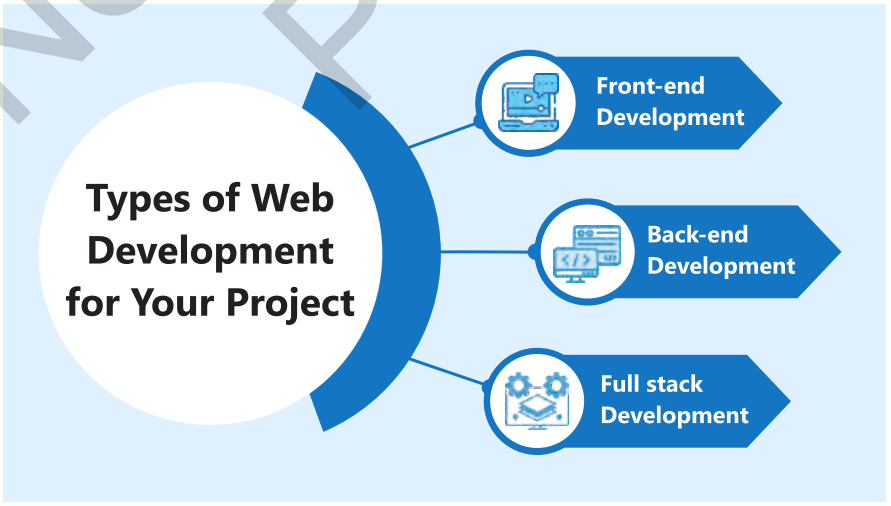


Figure 8.1: Types of Web Development

Example: Login System

A login system is a common feature in web development, allowing users to access their accounts on a website. This example will illustrate the roles of front-end and back-end development, as well as the concept of full-stack development.

3. Full-Stack Development

In the case of login system, a full-stack developer will create the User Interface (UI) for front-end and handle user authentication and database interaction for back-end.

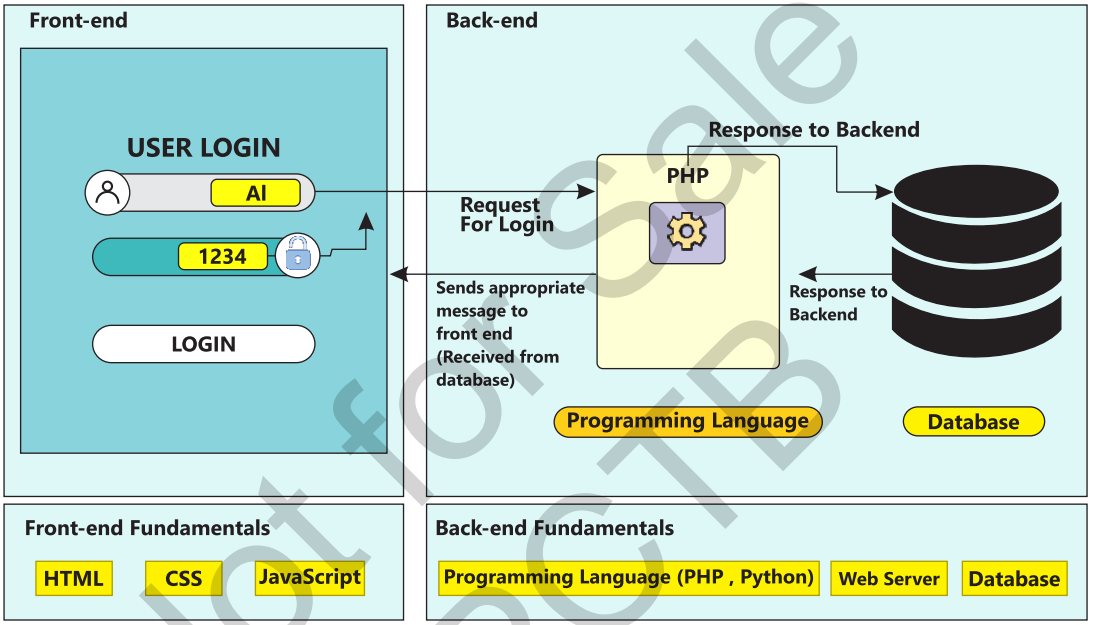
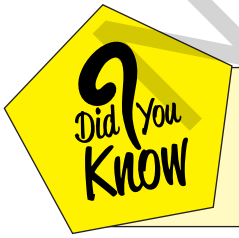


Figure 8.2: Graphical abstract of login system



Full-stack developers are in high demand because they can manage and develop all aspects of a web application, making them versatile and valuable in the tech industry.

8.3 Getting Started with HTML

HTML is the standard language used to create web pages. Think of HTML as the building blocks of a website. Just like LEGO pieces (as shown in Figure 8.3) come together to build a structure, HTML tags come together to build a web page.

Figure 8.3: LEGO Pieces

8.3.1 History of HTML

HTML was created by Tim Berners-Lee in 1991. It was designed to make sharing of information on the internet easy. Over the years, HTML has gone through many changes and improvements to make it more powerful and easier to use.

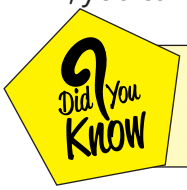
- **HTML 1.0 (1991):** The very first version of HTML. It was simple and had basic features to create text and links.
- **HTML 2.0 (1995):** Introduced more tags and features, allowing for creating more complex web pages.
- **HTML 3.2 (1997):** Added new tags for creating tables, scripts, and applets.
- **HTML 4.0 (1997):** Brought major improvements, including support for multimedia elements like images and videos.
- **HTML 4.01 (1999):** Minor improvements in version 4.0
- **HTML5 (2014):** The latest version of HTML. It includes new elements for better multimedia support, graphics, and more interactive web pages.

8.3.2 Setting up a Development Environment

To start creating websites, you need a few basic tools and environments:

- **Text Editor:** This is where you write your HTML code. Popular text editors include Notepad++, Sublime Text, and Visual Studio Code.
- **Web Browser:** You will use this to view and test your HTML files. Common web browsers are Google Chrome, Mozilla Firefox, and Microsoft Edge.

Start with a simple text editor and a web browser. Once you are comfortable with HTML, you can explore more advanced tools.



Start with a simple text editor and a web browser. Once you are comfortable with HTML, you can explore more advanced tools.

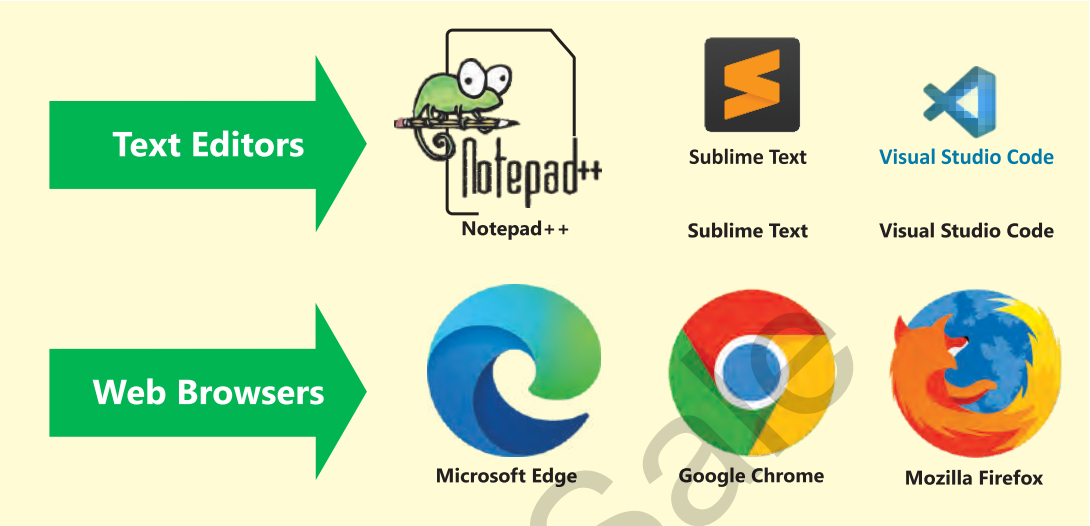


Figure 8.4: Fundamentals of website development environment

8.3.3 Creating a "Hello, World!" HTML Application

To create a basic HTML application that displays "a message" on a web page, follow these simple steps:

8.3.3.1 Writing the HTML Code

1. **Open your text editor,** You can use Notepad, Notepad++, Sublime Text, or any other text editor.
2. **Write the following HTML code** into your text editor.
3. **Save your file** with a html extension, for example, My_first_website.html.

HTML Code Example

```
<!DOCTYPE html>
<html>
    <head>
        <title>My First Web Page</title>
    </head>
    <body>
        <h1>Welcome to My Website</h1>
        <p>This is my first web page. I am learning HTML in the 9th
        class!</p>
</body>
</html>
```

8.3.3.2 Viewing the HTML File

1. Open Your Web browser (Google Chrome, Mozilla Firefox, and others).
2. Double-click on your file named My_first_website.html.
3. You should see the text welcome to my website displayed on the web page as shown in Figure 8.5.

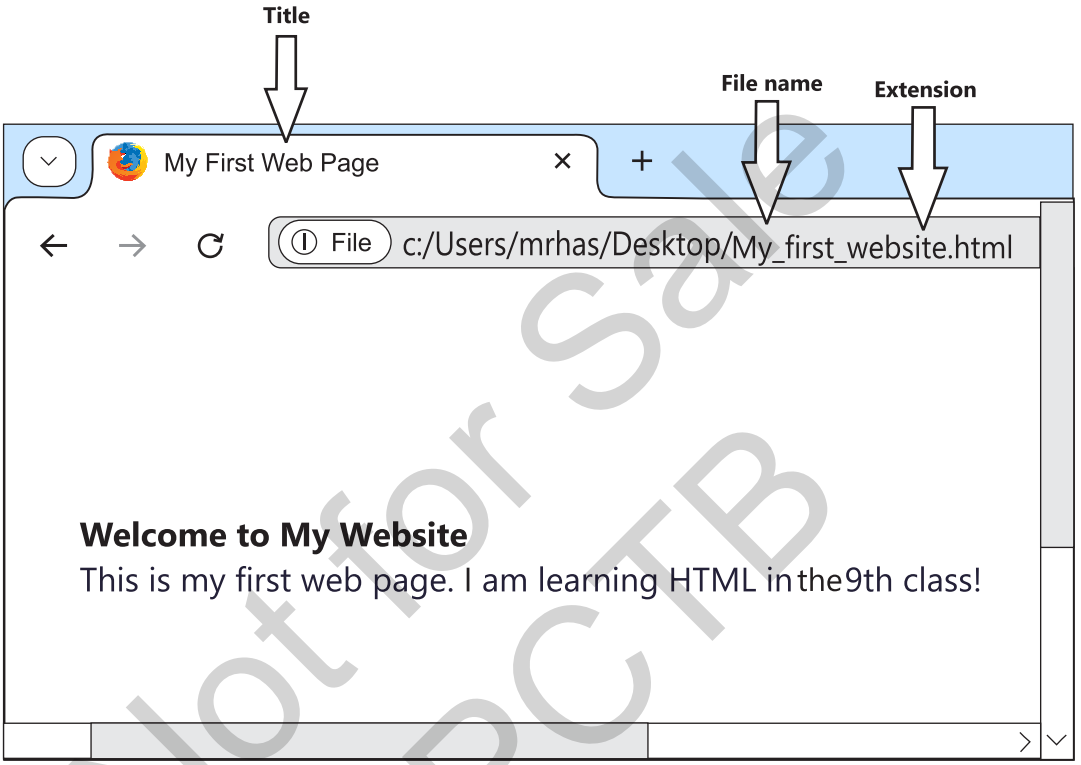
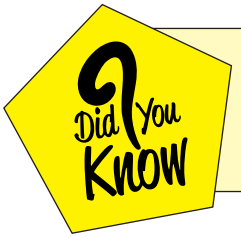


Figure 8.5: HTML in Web Browser

Tidbits

When saving your HTML files, always remember to add the .html extension at the end of your file name. This helps your computer recognize it as a web page. If you are using Notepad, make sure to select "All Files" in the "Save as type" dropdown, then type your file name followed by .html (e.g., hello_world.html).



If you make any changes to the HTML file, refresh the web page in your browser to see the updated content.

8.4 HTML Basic Structure

A structured HTML document is easier to read and understand. Properly nested and well-organized elements help developers and browsers interpret the content correctly. This organization ensures that the web page displays as intended.

Every HTML document has a basic structure where:

- <!DOCTYPE html>: This line tells the browser that this is an HTML5 document.
- <html>: This is the root element of an HTML page.
- <head>: This section contains meta-information about the HTML document, like the title.
- <title>: This sets the title of the web page, which appears in the browser tab.
- <body>: This section contains the content of the web page that you see in the browser.
- <h1>: This defines a large heading.
- : This defines a paragraph.

8.4.1 HTML Tags

Elements that make up an HTML document are called tags. A web page's structure and content are defined by them. On the basis of structure, HTML Tags are categorized into two types:

1. **Paired tags:** Comes in pairs an opening Tag and closing Tag i.e ..... .
2. **Unpaired Tags:** Do not need closing Tags. They are also known as self-closing Tags i.e., <img>,
.

8.5 Creating Content with HTML

Content in HTML is the main information on a web page that users read and interact with. It includes text, images, videos, links, and other elements that convey the purpose and message of the page. This makes it easier for people to find your site.

8.5.1 Headings

Headings in HTML, ranging from <h1> to <h6>, are used to define the structure and hierarchy of content on a web page. Here's why they are important:

Importance of Headings

1. **Organizing Content Headings** helps organize the content into sections and subsections, making it easier for users to read and understand. <h1> is typically used for the main title of the page, while <h2> to <h6> are used for subheadings in decreasing order of importance.

2. Search Engine Optimization (SEO): Search engines use headings to understand the structure and main topics of a web page. Proper use of headings can improve the page's SEO, helping it rank higher in search results.

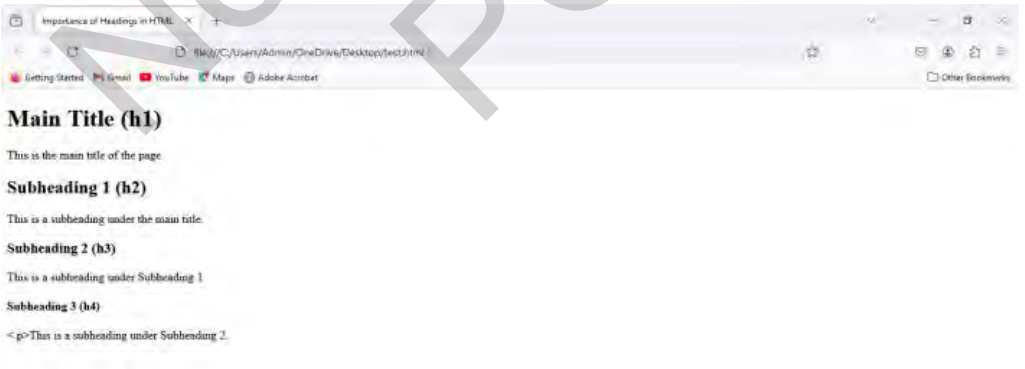
3. Consistent Formatting: Using standard heading tags ensures consistent formatting across different browsers and devices.

8.5.1.1 Example

Here is an example of how different heading levels can be used to organize content in a hierarchical structure:

```
<!DOCTYPE html>
<html>
    <head>
            <title>Importance of Headings in HTML</title>
    </head>
<body>
            <h1>Main Title (h1)</h1>
            <p>This is the main title of the page.</p>
            <h2>Subheading 1 (h2)</h2>
            <p>This is a subheading under the main title.</p>
            <h3>Subheading 2 (h3)</h3>
            <p>This is a subheading under Subheading 1.</p>
    </body>
</html>
```

Output



8.5.2 Paragraphs

Paragraphs in HTML are used to organize and separate text into readable sections. Each paragraph creates a block of text with space above and below it, making the content easier to read. Paragraphs starts with the <p> tag and it ends with <\p>

8.5.3 Links

Links in HTML are used to connect one web page to another. They allow you to click on words or images to go to different parts of the same web page or to other web pages on the internet.

Links are created using the <a> tag.

```
<a href="https://www.example.com">Visit Example.com</a>
<a href="mailto:example@example.com">Send Email</a>
```

Sometimes, links in HTML can also let you click to send an email. These special links start with mailto: and when you click them, they open your email program so you can send a message to the email address in the link.

8.5.4 Images

Images are important in HTML because they make web pages more attractive and engaging. Additionally, using images helps with branding, as logos and specific visuals make it easier for users to recognize a brand. Lastly, including alternate text for images ensures that visually impaired users can understand what the images represent.

Images are added using the <img> tag.

```
<img src="image.jpg" alt="A description II of the I___Iimage ">"
```

8.5.5 Lists

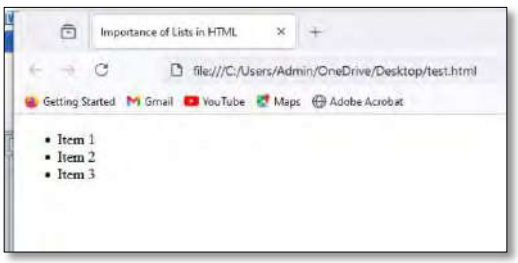
Lists improve readability by breaking complex ideas into simpler parts, allowing users to scan for details easily. Overall, lists make the content more organized and accessible for everyone. You can create ordered (numbered) and unordered (bulleted) lists.

8.5.5.1 Unordered List

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

Result

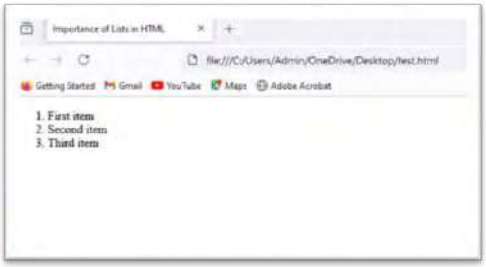
Output can be given side by



8.5.5.2 Ordered List

```
<ol>
    <li >First item</ li>
    <l i >Second item</ li>
    <l i >3rd item</ li>
</ ol>
```

Result



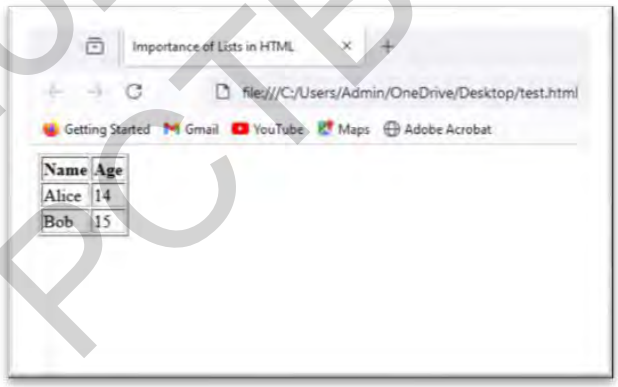
8.5.6 Creating Tables in HTML

Tables in HTML are used to display data in a structured format, allowing for easy comparison and organization of information. A table is created using the <table> tag, which contains rows defined by <tr> (table row) tags, and each row consists of cells represented by <td> (table data) tags. Additionally, headings for the table can be added using <th> (table header) tags to provide context for the data.

Example:

```
<table>
        <tr>
                <th>Name</th>
                <th>Age</th>
        </tr>
        <tr>
                <td>Alice</td>
                <td>14</td>
        </tr>
        <tr>
                <td>Bob</td>
                <td>15</td>
        </tr>
</table>
```

Output



8.5.7 HTML Comments

In HTML, comments can be extremely useful for:

- Explaining the purpose of a specific section of code
- Leaving reminders for future edits
- Temporarily disabling code for testing purposes

Syntax of HTML Comments

HTML comments begin with <!-- and end with -->. Any text placed within these markers will be treated as a comment and will not be rendered by the browser.

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

8.6 Styling with CSS

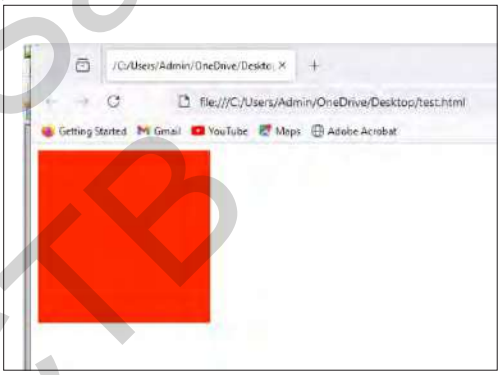
Styling with Cascading Style Sheets (CSS) is very important for improving the visual appearance of webpages and improving user experience. CSS allows web developers to control the colors, fonts, layout, and overall design of HTML elements, separating the content from the presentation. CSS offers various properties and selectors to apply styles to specific elements, enabling responsive design that automatically adjusts to different screen sizes and devices.

8.6.1 Basic Structure of CSS

The basic structure of CSS is essential for applying styles to HTML elements effectively. CSS is composed of rules that consist of selectors and declarations. Selectors specify which HTML elements the styles will apply to, while declarations define the specific styles to be applied, including properties and their corresponding values. A typical CSS rule follows this format:

selector { property: value;

}



For example, a simple CSS rule can changes the color and size of all headings on a web page:

```
h1 {
    color: red;
    font-size: 24px;
}
```

In this example, the CSS rule targets all '<h1>' elements, setting their text color to red and font size to 24 pixels.

8.6.2 Integrating CSS in HTML

Integrating CSS with HTML is essential for styling web pages and it can be done in three primary ways: inline, internal, and external styles.

1. **Inline Styles:** This method involves adding CSS directly to individual HTML elements using the style attribute. For example, <h1 style="color: blue;">Hello World</h1> changes the color of the heading to blue. While easy for quick changes, inline styles can make the code cluttered and less maintainable.
2. **Internal Styles:** CSS can also be included in the <head> section of an HTML document using the <style> tag. This method allows you to define styles for the entire page without affecting others.

For instance:

<style> h1 {

color: yellow;

}

</style>

3. External Styles: The most efficient method for larger projects is to use an external CSS file, which is linked to the HTML document with the <link> tag in the <head> section. This keeps the HTML clean and allows for easy updates across multiple pages.

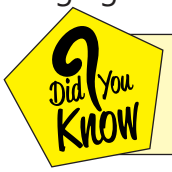
For example:

<link rel="stylesheet" href="styles.css">

By integrating CSS in these ways, developers can create visually appealing and well-organized web pages that enhance user experience.

8.6.4 Styling HTML Elements with Fonts, Colors, Backgrounds Styling Fonts

You can change the appearance of text on a web page using CSS. This includes changing the font family, size, weight, and style.



You can use different fonts by specifying their names in CSS. For example, you can set the font to Arial or Times New Roman.

Example of Styling Fonts

Here is how you can style the font of a paragraph:

P {

font-family: Arial, sans-serif;

font-size: 16px;

font-weight: bold;

font-style: italic;

}

In this example:

- **font-family:** Arial, sans-serif; sets the font to Arial. If Arial is not available, it will use a generic sans-serif font.
- **font-size:** 16px; sets the font size to 16 pixels.
- **font-weight:** bold; makes the text bold.
- **font-style:** italic; makes the text italic.

8.6.3.1 Styling Backgrounds

CSS is a language used to style web pages. One of the fun that you can do with CSS is to change the background of a web page or elements on it. Here are a few

ways you can style backgrounds of a web page with CSS:

- **Background Color:** You can change the background color of a web page or any HTML element using the background-color property. For example, if you want a blue background, you can write:
  body { background-color: blue;
  }
- **Background Image:** You can set an image as the background of a web page using the background-image property. For example, font family issue as the background, you can write:
  body {
  background-image: url("your-image.jpg");
  }
- **Background Repeat:** Sometimes, the background image is small, and you want it to repeat across the page. You can use the background-repeat property to do this. For example:
  body {
  background-image: url("your-image.jpg"); background-repeat: repeat;
  }
- **Background Position:** You can position the background image exactly where you want it using the background-position property. For example, to center the image, you can write:
  body {
  background-image: url("your-image.jpg"); background-position: center;
  }
- **Background Size:** You can control the size of the background image using the background-size property. For example, to cover the entire page with the background image, you can write:
  body {
  background-image: url("your-image.jpg"); background-size: cover;
  }

Using these properties, you can create colorful and appealing backgrounds for web pages.

8.6.4 Creating Layouts and Organizing Content

Creating layouts and organizing content on a web page is an important part of web design. CSS helps you arrange different parts of your web page in an organized way. Here are some basic methods to create layouts and organize content:

• **Divs and Sections:** HTML elements like and <section> are used to group content together. You can then use CSS to style and position them.

For example:

```
<div class="container">
    <section class="header">This is the header</section>
    <section class="content">This is the main content</section>
    <section class="footer">This is the footer</section>
</div>
```

• **CSS Grid:** The CSS Grid Layout is a powerful tool for creating complex layouts. It allows you to arrange items into rows and columns. For example:

```
    .container {
    display: grid;
    grid-template-columns: auto auto;
    grid-gap: 10px;
}
.item
{   padding: 20px;
    background-color: lightgrey;
}
```

• **CSS Flexbox:** Flexbox is another layout tool that helps in arranging items in a flexible and responsive way. It is useful for aligning items in a row or column.

For example:

```
.container
{   display: flex;
    justify-content: space-between;
}
.item
{   padding: 20px;
    background-color: lightgrey;
}
```

• **Positioning:** CSS positioning properties like position, top, left, right, and bottom allow you to place elements exactly where you want them on the webpage. **For example:**

```
.box
{   position: absolute;
    top: 50px;
    left: 100px;
    width: 200px;
    height: 100px;
    background-color: lightblue;
}
```

• **Margins and Padding:** Margins and padding are used to create space around

and inside elements. Margins create space outside the element, while padding creates space inside the element.

For example:

```
.box
{   margin: 20px;
    padding: 10px;
    background-color: lightgrey;
}
```

Tidbits

Always test your web page in different browsers to ensure that it looks and works the same way everywhere. This helps you catch any browser-specific issues early.

Class activity

In this activity, you will practice creating a basic web page layout using HTML and CSS. Follow these steps:

1. Create a new HTML file and name it "index.html".
2. Add the basic structure of an HTML document.
3. Inside the <body> tag, create a with the class name "container".
4. Inside the , add three sections with class names "header", "content", and "footer".
5. Link the CSS file to your HTML file using the <link> tag in the <head> section.
6. Use CSS to style the .container class with a grid layout, and apply background colors to the .header, .content, and .footer sections.
7. Test your web page in a browser to see your layout

8.6.5 Adding Animations and Transitions Using CSS

CSS animations and transitions can make your web pages more engaging by adding movement and effects. Let us learn how to use them!

8.6.5.1 Adding Animations

CSS allows you to add animations to your web page to make it more interactive. Animations can change the way elements look or move over a period of time. Here are some basic steps to create animations with CSS:

• **Define Keyframes:** Keyframes are used to specify the start and end points of an animation, as well as any intermediate steps. For example:

```
@keyframes example {
from {background-color: red;} to {background-color: yellow;}
}
```

This keyframe animation changes the background color from red to yellow.

• **Apply the Animation:** To apply the animation to an element, use the animation property.

For example:

```
.animated-box
{   width: 100px;
    height: 100px;
    background-color: red;
    animation-name: example;
    animation-duration: 4s;

}
```

This will change the background color of the box change from red to yellow for four seconds.

• **Loop and Timing:** You can also set how many times the animation should repeat and its timing function. For example:

```
.animated-box {
animation-iteration-count: infinite; /* Animation will repeat forever */-animation-timing-function: linear; * /Animation will progress at a constant speed */
}
```

8.6.5.2 Adding Transitions

CSS allows you to add transitions to a web page to make changes between styles smooth and visually appealing. Transitions can change properties like color, size, or position gradually, instead of instantly. Here are some basic steps to create transitions with CSS:

• Set the Initial Style: First, define the initial style for the element you want to animate. For example:

```
.box {
```

width: 100px;
height: 100px;
background-color: red;
transition: background-color 2s, width 2s;
}

This sets the initial size and color of the box, and specifies that changes to the background color and width should transition over 2 seconds.

• **Define the Hover State:** Next, define the styles for the element when it is hovered over. For example:

.box:hover {
background-color: yellow; width: 200px;
}

This will change the background color to yellow and double the width of the box when the mouse hovers over it.

8.7 Introduction to JavaScript

JavaScript is a programming language that is used to make websites interactive and engaging. It allows developers to create things like animations, games, and responsive features that react when you click buttons or move your mouse. For example, when you see a pop-up message on a web page or when an image changes when you hover over it, that's JavaScript at work. Execution of JavaScript in a flowchart is shown in Figure 8.7.

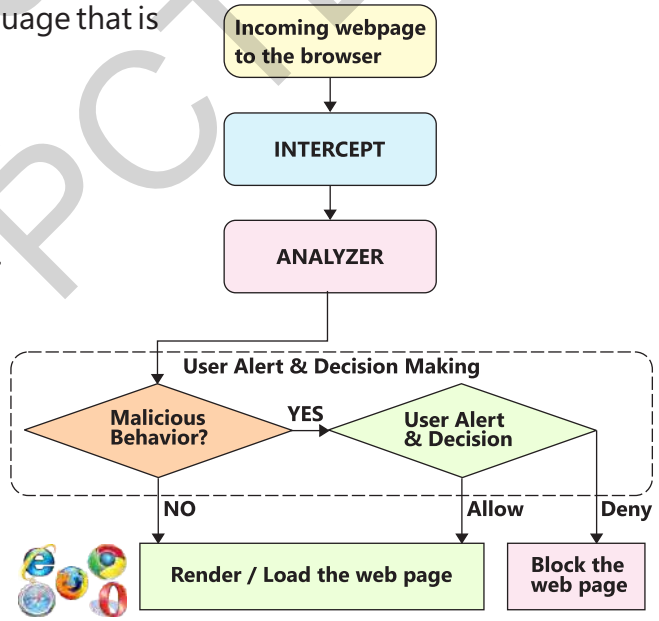
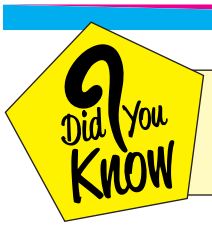


Figure 8.7: JavaScript Execution



JavaScript was created in just 10 days by Brendan Eich in 1995. It was initially called Mocha, then LiveScript, and finally JavaScript.

8.7.1 Basic Syntax and Examples

Here is a simple example to display an alert message using JavaScript:

```
<!DOCTYPE html>
<html>
        <head>
        <title>JavaScript Example</title>
        </head>
<body>
        <h1>Welcome to JavaScript</h1>
        <script>
        alert("Hello, 9th Class Students!");
        </script>
</body>
</html>
```

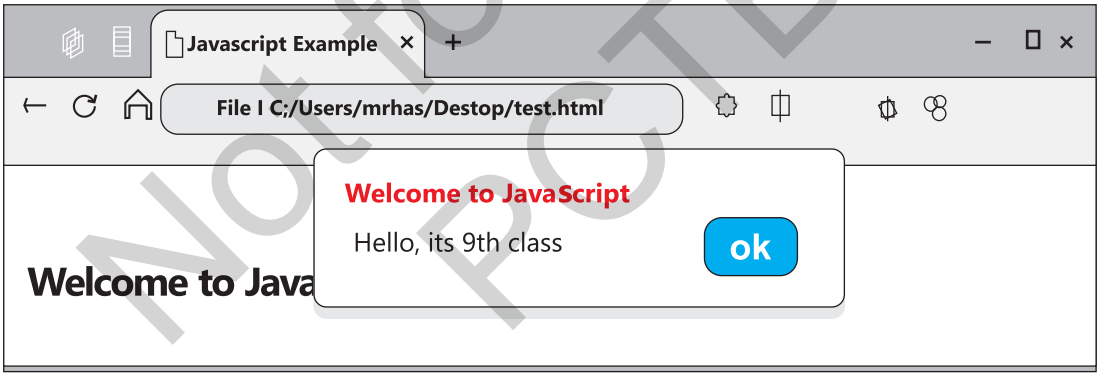


Figure 8.8: Alert Message Example

Class activity

Activity: Displaying an Alert

1. Open a text editor (e.g., Notepad).
2. Write the HTML and JavaScript code as shown above.
3. Save the file as "index.html".
4. Open the file in a web browser to see the alert message.

8.7.1 Variables and Data Types

In JavaScript, you can store data using variables. A variable is like a container that holds information which can be used and manipulated in your code.

8.7.1.1 Declaring Variables

To declare a variable in JavaScript, you use the var, let, or const keyword. Here's an example using var:

```
<script>
var name = "Athar";
var age = 15;
alert("Name: " + name +," Age: " + age);
</script>
```

8.7.1.2 Dry Run Example

Let's dry-run the above script to understand how it works step-by-step:

1. Declare variables: name = "Athar"; and age = 15;
2. Display alert: alert("Name: " + name + ", Age: "+ age);

This will result in an alert box showing: Name: Athar, Age: 15.

8.7.1.3 Data Types

Variables can store different types of data. Here are some common data types in JavaScript:

String:

A sequence of characters used for text.

var name = "Athar"; // String

Number:

Represents both integer and floating-point numbers.

var age = 15; // Number

Boolean:

Represents true or false values.

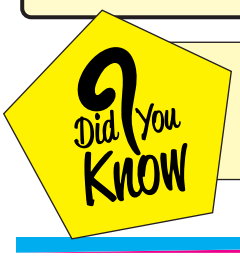
var isStudent = true; // Boolean

Array:

A collection of values stored in a single variable. var scores = [90, 85, 88]; // Array

Tidbits

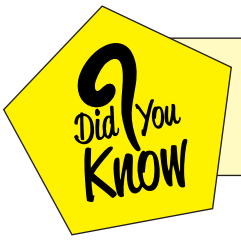
Always use quotes for strings and no quotes for numbers and booleans.



You can change the value of a variable after it has been declared. For example, age = 16; will change the value of age to 16.

Class activity

Create your own variables with different data types. For example, create a variable for your favorite color, your current grade, or whether you like coding.



JavaScript is the most popular programming language in the world. It is used by 95.2% of all websites, as of 2021.

8.7.2 Functions in JavaScript

Functions allow you to reuse code and perform specific tasks. They are like mini-programs that you can run whenever you need them. Let's learn how to create and use functions in JavaScript.

8.7.2.1 Simple Function

Here's an example of a simple function that displays a greeting message:

```
<script>
function greet() {
alert("Hello, Student!");
}
greet(); // This calls the function to execute
</script>
```

In this example:

- The function greet() declares a function named greet.
- alert("Hello, Student!"); is the code that runs when the function is called.
- greet(); calls the function, displaying the alert message.

8.7.2.2 Function with Parameters

Sometimes, you want your function to do something with input values. You can achieve this using parameters.

8.7.2.3 Function with Multiple Parameters

You can also create functions that take multiple parameters. Here's an example:

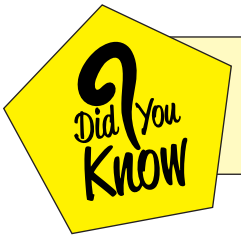
```
<script>
function addNumbers(a, b) {
var sum = a + b;
alert("The sum is: " + sum);
}
addNumbers(5, 3); // This calls the function with the parameters 5 and 3
</script>
```

Tidbits

Parameters are placeholders for values that you pass to the function. You can use these values inside your function.

In this example:

- The function addNumbers(a, b) declares a function that takes two parameters, a and b.
- var sum = a + b; calculates the sum of a and b.
- alert('The sum is: ' + sum); displays the result of the addition.
- addNumbers(5, 3); calls the function with the arguments 5 and 3, resulting in the alert message "The sum is: 8".



Functions can take any number of parameters. You can pass different values each time you call the function.

Creating a Function **Class activity**

1. Write a function that takes a name as a parameter and displays a personalized greeting. Call the function with your name to see the greeting.
2. Write a function that calculates the area of a rectangle given its length and width.

8.7.3 Handling Events and User Input

JavaScript allows you to make your web page interactive by handling events and user input. An event is an action that occurs when a user interacts with a webpage, like clicking a button or pressing a key.

HTML Events

HTML events are actions that occur in the browser, often triggered by user interactions. Events can be used to make web pages interactive by executing JavaScript code when a specific event occurs.

Common HTML Events

Here are some common events you might encounter:

- **onclick:** Triggered when an element is clicked.
- **onload:** Triggered when a page or an image has finished loading.
- **onmouseover:** Triggered when the mouse pointer moves over an element.
- **onmouseout:** Triggered when the mouse pointer moves out of an element.
- **onkeyup:** Triggered when a key is released on the keyboard.

8.7.3.1 Managing Events and User Interactions with JavaScript

Let us learn how to manage events and user interactions step-by-step.

Event Handlers

An event handler is a function that runs when a specific event occurs. You can attach event handlers to HTML elements to make them respond to user actions.

Example: Button Click Event

Here's an example of how to handle a button-click event:

```
<!DOCTYPE html>
<html >
    <head>
        <title>Button Click Event</title>
        <script>
        function showMessage() {
        alert("Button was clicked!");
}
        </script>
    </head>
<body>
        <button onclick="showMessage()">Click Me!</button>
</body>
</html>
```

Tidbits

Use descriptive names for your functions to make your code easier to understand.

8.7.4 Creating Interactive Elements

In this section, we will learn how to make web pages interactive by developing simple programs and forms. We will also learn how to integrate JavaScript with HTML to add interactive functionality.

8.7.4.1 Developing Simple Programs and Forms

Forms allow users to input data, which can be processed using JavaScript. Here is an example of a simple form that takes a user's name and displays a greeting message.

Example: Simple Form

Create an "index.html" file with the following content:

```
<!DOCTYPE html>
<html>
<head>
    <title>Interactive Form</title>
    <script>
        function greetUser() {
```

```
        var name = document.getElementById('name').value;
        alert("Hello, " + name + "!");
    }
  </script>
</head>
<body>
  <h1>Welcome!</h1>
  <form>
    <label for="name">Enter your name:</label>
    <input type="text" id="name" name="name">
    <button type="button" onclick="greetUser()">Submit</button>
  </form>
</body>
</html>
```

In this example:

- The form element contains an input field for the user to enter their name and a button to submit the form.
- The onclick attribute of the button calls the greetuser () function when the button is clicked.
- The greetuser () function gets the value of the input field and displays an alert with a greeting message.

8.7.4.2 Integrating JavaScript with HTML for Interactive Functionality

JavaScript can be used to add interactive functionality to HTML elements. Let's create a simple program that changes the background color of the page when a button is clicked.

Example: Changing Background Color

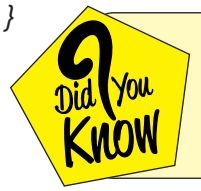
Add the following content to your index.html file:

```
<!DOCTYPE html>
<html>
    <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Change Background Color</title>
```

```
    <script src="script.js"></script>
    </head>
    <body>
    <h1>Welcome to the Color Changer!</h1>
    <button onclick="changeColor()">Change Background Color</button>
    </body>
</html>
```

Add the following content to your script.js file:

```
function changeColor() {
document.body.style.backgroundColor = "lightblue";
}
```



You can use JavaScript to change any CSS property of an HTML element, such as color, font-size, or visibility.

Class activity

Create a button that hides an element on the page when clicked. Experiment with different JavaScript functions and HTML elements.

By combining JavaScript with HTML, you can create interactive and dynamic web pages that respond to user input and actions. Keep practicing to discover more ways to make your web pages engaging and interactive.

8.8 Developing and Debugging

Testing and debugging are important steps in web development. They help you find and fix errors in your code to ensure your web pages work correctly.

8.8.1 Debugging Techniques

Debugging is the process of finding and fixing issues in your code. Here are some common debugging techniques:

1. Using Browser Developer Tools

Most web browsers have built-in developer tools that help you debug your code. For example, you can use the console to see error messages and set break points to pause your code and examine its behavior.

```
<script>
    console.log("This is a debug message");
    var x = 10;
    console.log("The value of x is: " + x);
```

</script>

2. Reading Error Messages

When something goes wrong, your browser will usually display an error message. Reading these messages carefully can help you understand what went wrong and how to fix it.

3. Checking Your Code

Go through your code line by line to check for common issues like missing semicolons, unmatched braces, or incorrect variable names.

8.8.2 Identifying and Fixing Common Issues

Here are some common issues in web development and how to fix them:

1. Broken Links

Make sure all your links point to the correct URLs. Double-check the paths to your files.

2. Incorrect HTML Structure

Ensure your HTML tags are properly nested and closed.

3. CSS Issues

Verify that your CSS selectors are correct and that there are no typos in your styles.

8.8.3 Deploying and Testing

After developing your web page, it's important to test it across different browsers and devices to make sure it works for all users.

8.8.3.1 Strategies for Testing Web Pages

1. Cross-Browser Testing

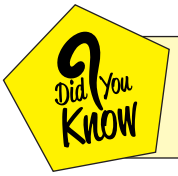
Web pages can look different in different browsers. Test your web page in multiple browsers like Chrome, Firefox, and Edge to ensure consistency.

2. Responsive Design Testing

Make sure your web page looks good on all devices, including desktops, tablets, and smartphones. Use tools like the browser's responsive design mode to test how your page looks on different screen sizes.

3. User Testing

Ask your friends or family members to use your web page and provide feedback. They may find issues that you missed.



You can use the browser's inspector tool to see how your CSS is applied to elements.

Summary

- Web development is the process of creating websites and web applications. It involves using various programming languages and tools to design, build, and maintain websites.
- Front-end Development focuses on what users see and interact with on a website.
- Back-end Development manages the behind-the-scenes part of a website, like servers, databases, and application logic.
- HTML stands for HyperText Markup Language. It's the standard language used to create web pages. Think of HTML as the building blocks of a website.
- Content in HTML is the main information on a web page that users read and interact with.
- Headings in HTML, ranging from <h1> to <h6>, are used to define the structure and hierarchy of content on a web page.
- Paragraphs in HTML are used to organize and separate text into readable sections.
- Links in HTML are used to connect one web page to another. They allow you click on words or images to go to different parts of the same page or to other pages on the internet.
- In HTML, comments are used to insert notes or explanations within the code.
- Styling with CSS (Cascading Style Sheets) is essential for enhancing the visual appearance of web pages and improving user experience.
- JavaScript is a programming language that is used to make websites interactive and engaging.
  Debugging is the process of finding and fixing issues in your code.

Multiple Choice Questions

1. Which of the following tag is not a correct HTML tag?

(a) (b) (c) <head> (d) <footer>

2. What does CSS stand for?

(a) Cascading Style Sheets (b) Computer Style Sheets
(c) Creative Style Sheets (d) Colorful Style Sheets

3. Which of the following tag is used to create a hyperlink in HTML?

(a) <link> (b) <a> (c) <href> (d) <nav>

4. Which property is used to change the background color in CSS?

(a) color (b) background-color
(c) bgcolor (d) background

5. Which HTML attribute is used to define inline styles?

(a) class (b) style (c) font (d) styles

6. Which of the following is the correct syntax for a CSS rule?

(a) selector {property: value;} (b) selector: {property=value;}
(c) selector {property=value} (d) selector: {property: value;}

7. In JavaScript, which markup is used for comments?

(a) /* */ (b) // (c) <– (d) /* */

8. How do you include JavaScript in an HTML document?

(a) <script src="script.js"></script> (b) <java src="script.js"></java>
(c) <js src="script.js"></js> (d) <code src="script.js"></code>

9. Which HTML tag is used to create an unordered list?

(a) (b) (c) (d) <list>

10. Which tag is used to display a horizontal line in HTML?

(a)
 (b) <hr> (c) <line> (d) <hline>

Short Questions

1. What is the purpose of the <head> tag in HTML?
2. Explain the difference between an ordered list and an unordered list in HTML.
3. How do you add a comment in CSS?
4. What are the different ways to apply CSS to an HTML document?
5. How can you include JavaScript in an HTML file?
6. Describe the syntax for creating a hyperlink in HTML.
7. What is the function of the tag in HTML?
8. How do you link an external CSS file to an HTML document?
9. What is the use of the <table> tag in HTML?
10. Explain the box model in CSS.

Long Questions

1. Discuss the fundamental differences between HTML, CSS, and JavaScript in the context of web development.
2. Explain the process of setting up a development environment for web development. By disscussing the necessary softwares and tools.
3. Create a basic HTML page that includes a header, a paragraph, an image, and a hyperlink.
4. How do you style a table using CSS? Create a sample table and apply styles to it.
5. Describe the different CSS selectors and provide examples of each.
6. Explain the process of creating a responsive web page using CSS with the help of examples and explanations.
7. Write a JavaScript function that changes the background color of a web page when a button is clicked. Provide the complete code and explain how it works.
8. How do you add animations and transitions using CSS? Provide examples and explain the properties involved.