HTML5

TABLE OF CONTENTS

Preface 2
Introduction 2
Basic HTML5 Syntax 2
- Document Structure 2
- Head Section 3
- Body Section 4

HTML5 Elements 5
- Formatting Elements 5
- Link Elements 6
- Image Elements 7
- Table Elements 8
- Form Elements 9
- Multimedia Elements 10

HTML5 Attributes 10
- Global Attributes 10
- Event Attributes 11
- Form Attributes 11
- Input Attributes 12
- Link Attributes 13
- Media Attributes 14
- Table Attributes 14

HTML5 Forms 15
- Form Controls 15
- Form Validation 16

HTML5 Graphics 17
- Canvas 17
- SVG 17
- Audio Element 17
- Video Element 18
- Canvas and SVG Graphics with Javascript 18

HTML5 APIs 18
- Geolocation API 18
- Web Storage API 18
- Canvas API 18
- Web Audio API 18
- Drag and Drop API 19

Web Workers API 20
Best Practices 20
Accessibility 20
SEO 21
Performance 21
HTML5 Resources 22
References 22
Tools 22
Frameworks and Libraries 23

PREFACE

This HTML5 cheatsheet is a quick reference guide for web developers who want to quickly look up HTML5 syntax, elements, attributes, APIs, and best practices. It provides a comprehensive overview of HTML5, including its document structure, head and body sections, text, formatting, link, image, table, form, multimedia, and graphics elements. It also covers HTML5 APIs, such as the Drag and Drop API, as well as best practices for accessibility, SEO, and performance. In addition, this cheatsheet includes a list of helpful references, tools, frameworks, and libraries for web developers to use.

INTRODUCTION

HTML5 is the latest version of the HTML markup language used for creating web pages and applications. With its new features and improvements, HTML5 offers developers more flexibility and ease of use when creating web content. This HTML5 cheatsheet is designed to serve as a quick reference guide for web developers who want to create HTML5 compliant pages or applications. It includes sections on basic HTML5 syntax, document structure, elements, attributes, APIs, best practices, and resources. Each section includes examples and explanations to help you quickly understand the concepts and use them effectively in your own web development projects. Whether you are a beginner or an experienced developer, this cheatsheet will be a useful resource to keep at your fingertips.

BASIC HTML5 SYNTAX

DOCUMENT STRUCTURE

The document structure of an HTML5 document is the overall structure that defines how the document is organized and displayed in a web browser. The basic document structure of an HTML5 document consists of the following elements:

Element	Description
<!DOCTYPE>	This is the first element in an HTML5 document and specifies the version of HTML being used
<html>	This is the second element in an HTML5 document and wraps around all the other elements in the document. It has two main sections:a. The head section - This section contains meta data about the document, such as the title, keywords, and description, as well as links to external stylesheets and scripts.b. The body section - This section contains the actual content of the document, such as text, images, and other multimedia elements

```
<!DOCTYPE html>
<html>
  <head>
    <title>My HTML5 Document</title>
    <meta charset="UTF-8">
    <meta name="description"
content="This is my HTML5 document">
    <meta name="keywords"
content="HTML5, document, web">
    <link rel="stylesheet"
href="style.css">
  </head>
  <body>
    <h1>Welcome to My HTML5
Document</h1>
    <p>This is the content of my
HTML5 document.
    <img src="image.jpg" alt="An
image">
  </body>
</html>
</pre>
```

In this example, we have the <!DOCTYPE> declaration followed by the <html> element. Inside the <html> element, we have the <head> element that contains

the title, meta data, and stylesheet. The `<body>` element contains the actual content of the document, which includes a header, a paragraph of text, and an image.

HEAD SECTION

The head section of an HTML5 document is used to provide metadata about the document, including the document title, links to stylesheets, and scripts. The head section is not displayed on the page itself, but rather is used by the browser to provide information about the page.

The following elements can be included in the head section of an HTML5 document:

Element	Description
`<title>`	Defines the title of the document, which appears in the browser's title bar and is used as the default name for bookmarks
`<meta>`	Defines metadata about the document, including the character encoding, description, and keywords. Some common attributes for the `<meta>` element include: `charset`: Specifies the character encoding for the document. `name="description"`: Specifies a brief description of the document. `name="keywords"`: Specifies a list of comma-separated keywords for the document.
`<link>`	Defines a link to an external resource, such as a stylesheet or script. Some common attributes for the `<link>` element include: `rel`: Specifies the relationship between the current document and the linked resource. `type`: Specifies the MIME type of the linked resource. `href`: Specifies the URL of the linked resource.
`<script>`	Defines a script that is executed on the client-side. The script can be included directly in the head section, or it can be included in an external file using the `src` attribute. Some common attributes for the `<script>` element include: `type`: Specifies the MIME type of the script. `src`: Specifies the URL of the external script file. `async`: Specifies that the script should be executed asynchronously. `defer`: Specifies that the script should be executed after the page has finished loading.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>My Website</title>
    <meta name="description"
content="This is a description of my
website.">
    <link rel="stylesheet"
href="style.css">
  </head>
```

```
  <body>
    <!-- The rest of the HTML code
goes here -->
  </body>
</html>
</pre>
```

In this example, the `head` element contains several important elements for the web page, including:

- The `meta` element with the `charset` attribute, which sets the character encoding for the web page to UTF-8.
- The `title` element, which sets the title of the web page that appears in the browser tab or window.
- The `meta` element with the `description` attribute, which provides a brief description of the content of the web page for search engines and social media sites.
- The `link` element with the `rel` attribute set to "stylesheet", which links to an external CSS stylesheet file named "style.css". This file contains the styling information for the web page.

This is just a basic example, and there are many other elements and techniques that can be used in the head section of an HTML5 document, depending on the specific needs of the web page.

BODY SECTION

The `<body>` section of an HTML5 document contains the content that is displayed to the user. This is where you will include text, images, videos, and other elements that you want to appear on the page.

Some of the most commonly used HTML5 elements for the body section include:

Element	Description
`<h1>` to `<h6>`	These elements are used to create headings of different sizes. They are typically used to break up the content into sections and subsections, with `<h1>` being the largest and most important heading and `<h6>` being the smallest and least important
`<p>`	This element is used to create paragraphs of text. It is one of the most commonly used elements in the body section and is used to display large blocks of text
`<ul>`	This element is used to create unordered lists. It is typically used to display a list of items that do not need to be in any particular order
`<ol>`	This element is used to create ordered lists. It is typically used to display a list of items that need to be in a specific order
`<li>`	This element is used to create individual list items within a list
`<img>`	This element is used to display images on the page. It is typically used to display logos, photos, or other types of graphics
`<a>`	This element is used to create links to other pages or resources. It is typically used to create navigation links or to link to related content

Element	Description
	This element is used to create a container for other elements on the page. It is typically used to group related content together and apply styles to the group as a whole
	This element is used to apply styles to a small piece of text within a larger block of text
<form>	This element is used to create a form on the page. It is typically used to gather information from the user, such as their name, email address, or other details

```
<body>
  <header>
    <h1>My Website</h1>
    <nav>
      <ul>
        <li><a
href="#">Home</a></li>
        <li><a
href="#">About</a></li>
        <li><a
href="#">Contact</a></li>
      </ul>
    </nav>
  </header>

  <main>
    <article>
      <h2>Welcome to My Website</h2>
      <p>This is a paragraph of text
that describes my website.
    </article>

    <aside>
      <h3>Recent Posts</h3>
      <ul>
        <li><a href="#">Post
1</a></li>
        <li><a href="#">Post
2</a></li>
        <li><a href="#">Post
3</a></li>
      </ul>
    </aside>
  </main>

  <footer>
    &copy; 2023 My Website. All
rights reserved.
  </footer>
</body>
```

In this example, the `body` element contains three other elements: `header`, `main`, and `footer`. The `header` element contains the main heading for the web page (`h1`) and a navigation menu (`nav`) that links to different sections of the website.

The `main` element contains two other elements: `article` and `aside`. The `article` element contains a subheading (`h2`) and a paragraph of text, while the `aside` element contains a list of recent posts.

The `footer` element contains a small paragraph of text that includes the copyright symbol and the name of the website.

HTML5 ELEMENTS

FORMATTING ELEMENTS

HTML5 includes several formatting elements that allow you to control the appearance of text on a web page. These elements are used to apply styles such as bold, italic, and underlining to text, as well as to highlight specific words or phrases.

Some of the most commonly used formatting elements in HTML5 include:

Element	Description
<b>	This element is used to apply bold formatting to text on the page. It is often used to highlight important words or phrases

Element	Description
<i>	This element is used to apply italic formatting to text on the page. It is often used to indicate that a word or phrase is being emphasized
<u>	This element is used to apply underlining to text on the page. It is often used to highlight key words or phrases
<mark>	This element is used to highlight text on the page. It is often used to indicate that a word or phrase is important or significant
<small>	This element is used to reduce the size of text on the page. It is often used for fine print or disclaimers
<sup> <sub>	These elements are used to create superscript and subscript text on the page, respectively. They are often used in scientific or mathematical contexts
<code>	This element is used to display code snippets on the page. It is often used in technical documentation or programming tutorials
<pre>	This element is used to preserve whitespace and formatting within a block of text on the page. It is often used for displaying code or other types of preformatted text

```
<p>This is a paragraph with
<strong>strong</strong> and
<em>emphasized</em> text.
```

In this example, the `p` element is used to create a paragraph of text, and the `strong` and `em` elements are used to apply different styles to parts of the text. The `strong` element is used to indicate text that should be displayed in a bold font, while the `em` element is used to indicate text that should be displayed in an italic font.

Here's another example that demonstrates the use of the `mark` and `del` elements:

```
<p>My favorite color is
<mark>blue</mark>, but sometimes I
wear a <del>red</del> shirt just to
be different.
</pre>
```

In this example, the `mark` element is used to highlight the word "blue" in the text, while the `del` element is used to indicate that the word "red" has been deleted (i.e. it should be struck through).

LINK ELEMENTS

HTML5 provides several link elements that allow you to create links between web pages and other types of content. These elements include:

Element	Description
<a>	This element is used to apply bold formatting to text on the page. It is often used to highlight important words or phrases
<link>	This element is used to apply italic formatting to text on the page. It is often used to indicate that a word or phrase is being emphasized
<img>	This element is used to apply underlining to text on the page. It is often used to highlight key words or phrases

Element	Description
<audio>	This element is used to highlight text on the page. It is often used to indicate that a word or phrase is important or significant
<video>	This element is used to reduce the size of text on the page. It is often used for fine print or disclaimers
<iframe>	These elements are used to create superscript and subscript text on the page, respectively. They are often used in scientific or mathematical contexts

```
<head>
  <link rel="stylesheet"
href="path/to/stylesheet.css">
  <link rel="icon"
href="path/to/favicon.ico">
  <title>My Website</title>
</head>
</pre>
```

In this example, two link elements are used to include a stylesheet and a favicon on the web page. The `rel` attribute is used to specify the relationship between the current document and the linked document. The `href` attribute is used to specify the path to the linked document.

The first `link` element includes a CSS stylesheet, which will be used to style the content of the web page. The second `link` element includes a favicon, which is a small icon that appears in the browser tab next to the title of the web page.

Note that the `link` element is also used to link to other types of resources, such as feeds, scripts, and alternate versions of the web page for different devices.

IMAGE ELEMENTS

Images are a crucial component of most web pages, and HTML5 provides several elements that allow you to include and display images on your web pages. These elements include:

Element	Description
<img>	This element is used to embed images on the page. It can be used to display images that are stored locally or to link to images that are hosted on external servers. The <img> element has several attributes that allow you to specify the source of the image, its dimensions, and its alt text (which is displayed if the image cannot be loaded)
<picture>	This element is used to provide alternative versions of an image based on the user's device or screen size. It can be used to serve different images to desktop and mobile users, for example
<svg>	This element is used to embed scalable vector graphics (SVG) on the page. SVG images are resolution-independent and can be scaled to any size without losing clarity or quality
<canvas>	This element is used to create dynamic graphics and animations on the page using JavaScript code. It can be used to create custom images, charts, graphs, and other types of visual content

```
<img src="path/to/image.jpg" alt="A
beautiful mountain landscape.">
```

In this example, the `img` element is used to display an image on the web page. The `src` attribute specifies the path to the image file, and the `alt` attribute provides a text description of the image. If the image cannot be displayed for any reason (e.g. if the file is missing or the network connection is down), the text in the `alt` attribute will be displayed instead.

```
<img src="path/to/image.jpg" alt="A
beautiful mountain landscape."
width="500" height="300"
title="Click to view larger image">
```

In this example, the `width` and `height` attributes are used to specify the dimensions of the image. The `title` attribute is used to provide a tooltip that will be displayed when the user hovers over the image. When the user clicks on the image, they will be taken to a larger version of the image (assuming that a larger version is available).

TABLE ELEMENTS

HTML5 includes several elements that allow you to create tables on your web pages. These elements include:

Element	Description
`<table>`	This element is used to create the table itself. It can contain one or more `<tr>` (table row) elements
`<tr>`	This element is used to create a row in the table. It can contain one or more `<td>` (table data) elements
`<td>`	This element is used to create a cell in the table. It can contain any type of content, including text, images, links, and other elements
`<th>`	This element is used to create a header cell in the table. It is typically used in the first row of the table to label the columns
`<thead> <tbody> <tfoot>`	These elements are used to group table rows into different sections. The `<thead>` element is used to group the header rows, the `<tbody>` element is used to group the body rows, and the `<tfoot>` element is used to group the footer rows

```
<table>
  <thead>
    <tr>
      <th>Product Name</th>
      <th>Description</th>
      <th>Price</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Product A</td>
      <td>This is a great
product!</td>
      <td>$19.99</td>
    </tr>
    <tr>
      <td>Product B</td>
      <td>This is another great
product!</td>
      <td>$24.99</td>
    </tr>
  </tbody>
</table>
```

This code will create a simple table with three columns: Product Name, Description, and Price. The `thead` element defines the header row of the table, while the `tbody` element defines the body of the table. Each row of the table is defined by a `tr` element, and each cell in the row is defined by a `td`

element. The `th` element is used to define the header cells of the table.

FORM ELEMENTS

HTML5 includes several elements that allow you to create forms on your web pages. These elements include:

Element	Description
`<form>`	This element is used to create a form on the page. It requires the `action` attribute to specify where the form data should be submitted, and the `method` attribute to specify how the form data should be submitted (usually `get` or `post`)
`<input>`	This element is used to create form fields for users to enter data. It can have several different types, such as `text`, `email`, `password`, `checkbox`, `radio`, `submit`, and `reset`. The `name` attribute is used to identify the field when the form is submitted
`<label>`	This element is used to associate a label with a form field. The `for` attribute should match the `id` attribute of the form field
`<textarea>`	This element is used to create a larger text field for users to enter longer messages or comments
`<select> <option>`	These elements are used to create dropdown menus or selection lists. The `select` element creates the dropdown menu, while the `option` element creates each individual option within the menu

```
<form action="submit-form.php"
method="post">
  <label for="name">Name:</label>
  <input type="text" id="name"
name="name"><br>

  <label>Gender:</label>
  <input type="radio" id="male"
name="gender" value="male">
  <label for="male">Male</label>
  <input type="radio" id="female"
name="gender" value="female">
  <label
for="female">Female</label><br>

  <input type="checkbox"
id="newsletter" name="newsletter">
  <label for="newsletter">Sign up
for our newsletter</label><br>

  <label for="color">Favorite
color:</label>
  <select id="color" name="color">
    <option value="red">Red</option>
    <option
value="green">Green</option>
    <option
value="blue">Blue</option>
  </select><br>

  <input type="submit"
value="Submit">
</form>
```

This form will submit the data to a file called "submit-form.php" using the POST method when the user clicks the "Submit" button. The form

includes a text field for the user's name, a radio button group for the user's gender, a checkbox for the newsletter subscription, a dropdown menu for the user's favorite color, and a submit button to submit the form data.

MULTIMEDIA ELEMENTS

HTML5 provides several elements for embedding multimedia content into web pages. These elements include:

Element	Description
`<audio>`	This element is used to embed audio content into a web page. It can be used to play sound effects, background music, or even podcasts. You can specify the source of the audio file using the `src` attribute
`<video>`	This element is used to embed video content into a web page. It can be used to play videos, movie trailers, or even live streams. You can specify the source of the video file using the `src` attribute
`<img>`	This element is used to embed images into a web page. You can specify the source of the image file using the `src` attribute. You can also add an optional `alt` attribute to provide a description of the image for users who are visually impaired

Here's an example of how to use these elements to embed multimedia content into a web page:

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

<video controls width="640"
height="360">
  <source src="video/movie.mp4"
type="video/mp4">
  Your browser does not support the
video element.
</video>

<img src="images/photo.jpg" alt="A
beautiful mountain landscape.">
```

This code will embed an audio file called "music.mp3", a video file called "movie.mp4", and an image called "photo.jpg" into the web page. The `controls` attribute will add player controls to the audio and video elements, allowing users to play, pause, and adjust the volume. The `width` and `height` attributes will set the dimensions of the video player. Finally, the `alt` attribute will provide a description of the image for users who cannot see it.

HTML5 ATTRIBUTES

GLOBAL ATTRIBUTES

class attribute

This attribute is used to apply a class to an HTML element. Multiple elements can share the same class, and styling can be applied to all elements with that class using CSS. Here's an example:

```
<div class="container">
  <p class="text">Some text in a
paragraph.
```

id attribute

This attribute is used to assign a unique identifier to an HTML element. It is often used for styling or scripting purposes. Here's an example:

```
  <h1>My Website</h1>
```

style attribute

This attribute is used to apply inline styles to an HTML element. Inline styles override any styles defined in an external CSS file or in a `style` element in the head section of the HTML document. Here's an example:

```
<p style="color: red; font-size:
18px;">Some text in a paragraph with
inline styles.
```

title attribute

This attribute is used to provide additional information about an HTML element. It is often used to display a tooltip when the user hovers over the element with their mouse. Here's an example:

```
<img src="image.jpg" alt="An image"
title="This is a description of the
image.">
```

EVENT ATTRIBUTES

Sure, here are some examples of event attributes in HTML5:

onclick

This attribute is used to define a JavaScript function to be executed when the user clicks on an HTML element. Here's an example:

```
<button onclick="alert('Hello,
world!')">Click me!</button>
```

onload

This attribute is used to define a JavaScript function to be executed when an HTML page has finished loading. Here's an example:

```
<body onload="myFunction()">
  <!-- Rest of the HTML code goes
here -->
</body>

<script>
function myFunction() {
  alert("The page has finished
loading!");
}
</script>
```

onmouseover

This attribute is used to define a JavaScript function to be executed when the user moves their mouse over an HTML element. Here's an example:

```
Hover over me!

<script>
function changeColor(element) {
  element.style.backgroundColor =
"red";
}
</script>
```

onsubmit

This attribute is used to define a JavaScript function to be executed when an HTML form is submitted. Here's an example:

```
<form onsubmit="validateForm()">
  <!-- Form elements go here -->
  <button
type="submit">Submit</button>
</form>

<script>
function validateForm() {
  // Form validation code goes here
}
</script>
```

FORM ATTRIBUTES

action

This attribute is used to specify the URL of the server-side script that will process the form data

when the form is submitted. Here's an example:

```
<form action="process-form.php"
method="post">
  <!-- Form elements go here -->
  <button
type="submit">Submit</button>
</form>
```

method

This attribute is used to specify the HTTP method to be used when submitting the form. The two most common methods are `GET` and `POST`. Here's an example:

```
<form action="process-form.php"
method="post">
  <!-- Form elements go here -->
  <button
type="submit">Submit</button>
</form>
```

name

This attribute is used to give the form a name, which can be used to refer to the form in JavaScript code. Here's an example:

```
<form name="myForm" action="process-
form.php" method="post">
  <!-- Form elements go here -->
  <button
type="submit">Submit</button>
</form>

<script>
var myForm =
document.forms["myForm"];
// JavaScript code that interacts
with the form goes here
</script>
```

enctype

This attribute is used to specify the encoding type used when submitting the form. The two most common types are `application/x-www-form-urlencoded` (default) and `multipart/form-data`. Here's an example:

```
<form action="process-form.php"
method="post"
enctype="multipart/form-data">
  <!-- Form elements go here -->
  <button
type="submit">Submit</button>
</form>
```

INPUT ATTRIBUTES

type

This attribute is used to specify the type of input element. Some of the most common types are `text`, `password`, `checkbox`, `radio`, `submit`, `reset`, and `file`. Here are some examples:

```
<input type="text" name="username"
placeholder="Enter your username">
<input type="password"
name="password" placeholder="Enter
your password">
<input type="checkbox"
name="remember" id="remember"
value="true">
<label for="remember">Remember
me</label>
<input type="radio" name="gender"
id="male" value="male">
<label for="male">Male</label>
<input type="radio" name="gender"
id="female" value="female">
<label for="female">Female</label>
<input type="submit" value="Submit">
<input type="reset" value="Reset">
<input type="file" name="photo">
```

name

This attribute is used to give the input element a name, which can be used to refer to the input element in JavaScript code and to associate the

input element with a label element. Here's an example:

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

value

This attribute is used to specify the initial value of the input element. This is particularly useful for input types like `text` and `checkbox`. Here's an example:

```
<input type="text" name="username"
value="John">
<input type="checkbox"
name="remember" value="true"
checked>
```

required

This attribute is used to specify that the input element must be filled out before the form can be submitted. Here's an example:

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

placeholder

This attribute is used to provide a hint or example value for the input element. Here's an example:

```
<input type="text" name="email"
placeholder="Enter your email
address">
```

LINK ATTRIBUTES

href

This attribute is used to specify the URL of the linked resource, such as a web page, image, or file. Here's an example:

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

target

This attribute is used to specify where the linked resource will be displayed when the user clicks on the link. The value can be `_blank` to open the linked resource in a new window or tab, `_self` to open the linked resource in the same window or tab, or the name of a specific window or frame. Here's an example:

```
<a href="https://www.example.com"
target="_blank">Visit Example.com in
a new tab</a>
```

rel

This attribute is used to specify the relationship between the current document and the linked resource. The most common value is `stylesheet` to indicate that the linked resource is a CSS stylesheet that should be applied to the current document. Here's an example:

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

type

This attribute is used to specify the MIME type of the linked resource, such as `text/css` for a CSS stylesheet, `image/jpeg` for a JPEG image, or `application/pdf` for a PDF file. Here's an example:

```
<a href="document.pdf"
type="application/pdf">Download the
PDF document</a>
```

media

This attribute is used to specify the media type or

query for which the linked resource is intended. This is typically used with CSS stylesheets to apply different styles for different screen sizes or devices. Here's an example:

```
<link rel="stylesheet"
href="styles.css" media="(min-width:
768px)">
```

MEDIA ATTRIBUTES

autoplay

This attribute is used to automatically start playing a video or audio file when the page loads. Here's an example:

```
<video src="video.mp4"
autoplay></video>
```

controls

This attribute is used to display media controls, such as a play/pause button, volume control, and progress bar, for video and audio files. Here's an example:

```
<audio src="audio.mp3"
controls></audio>
```

loop

This attribute is used to loop a video or audio file continuously, so that it starts playing again from the beginning when it reaches the end. Here's an example:

```
<video src="video.mp4" loop></video>
```

poster

This attribute is used to specify an image to display as the poster frame for a video file, before it starts playing. Here's an example:

```
<video src="video.mp4"
poster="poster.jpg"></video>
```

preload

This attribute is used to specify whether and how a video or audio file should be preloaded, or loaded in the background, when the page loads. The possible values are `none`, `metadata`, or `auto`. Here's an example:

```
<audio src="audio.mp3"
preload="auto"></audio>
```

TABLE ATTRIBUTES

border

This attribute is used to specify the border size of a table in pixels. Here's an example:

```
<table border="1">
  <tr>
    <th>Header 1</th>
    <th>Header 2</th>
  </tr>
  <tr>
    <td>Row 1, Column 1</td>
    <td>Row 1, Column 2</td>
  </tr>
  <tr>
    <td>Row 2, Column 1</td>
    <td>Row 2, Column 2</td>
  </tr>
</table>
```

cellpadding

This attribute is used to specify the padding size of the cells within a table, in pixels. Here's an example:

```
<table cellpadding="5">
  <tr>
    <th>Header 1</th>
```

```
    <th>Header 2</th>
  </tr>
  <tr>
    <td>Row 1, Column 1</td>
    <td>Row 1, Column 2</td>
  </tr>
  <tr>
    <td>Row 2, Column 1</td>
    <td>Row 2, Column 2</td>
  </tr>
</table>
```

cellspacing

This attribute is used to specify the spacing size between the cells within a table, in pixels. Here's an example:

```
<table cellspacing="10">
  <tr>
    <th>Header 1</th>
    <th>Header 2</th>
  </tr>
  <tr>
    <td>Row 1, Column 1</td>
    <td>Row 1, Column 2</td>
  </tr>
  <tr>
    <td>Row 2, Column 1</td>
    <td>Row 2, Column 2</td>
  </tr>
</table>
```

width

This attribute is used to specify the width of a table, in pixels or percentage. Here's an example:

```
<table width="80%">
  <tr>
    <th>Header 1</th>
    <th>Header 2</th>
  </tr>
  <tr>
    <td>Row 1, Column 1</td>
    <td>Row 1, Column 2</td>
  </tr>
  <tr>
    <td>Row 2, Column 1</td>
    <td>Row 2, Column 2</td>
  </tr>
</table>
```

HTML5 FORMS

FORM CONTROLS

Form controls are the various types of input fields that can be used in HTML forms. Here are some examples of different form controls that can be used:

Text Input

The text input type allows the user to enter text data. It is the default input type.

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

Email Input

The email input type is used for email addresses. It ensures that the user enters a valid email address.

```
<label for="email">Email:</label>
<input type="email" id="email"
name="email" required>
```

Password Input

The password input type is used for password fields. It hides the user input and can help prevent unauthorized access.

```
<label
for="password">Password:</label>
<input type="password" id="password"
name="password" required>
```

Number Input

The number input type is used for numeric values. It provides additional options for specifying a minimum and maximum value.

```
<label for="age">Age:</label>
<input type="number" id="age"
name="age" min="18" max="100"
required>
```

Checkbox

The checkbox input type is used for a single checkbox that the user can select or deselect.

```
<label for="agree-to-terms">Agree to
terms and conditions:</label>
<input type="checkbox" id="agree-to-
terms" name="agree-to-terms"
required>
```

Radio Button

The radio button input type is used for a group of radio buttons that the user can select only one of the options.

```
<label for="gender-
male">Male</label>
<input type="radio" id="gender-male"
name="gender" value="male" required>

<label for="gender-
female">Female</label>
<input type="radio" id="gender-
female" name="gender" value="female"
required>
```

Select Box

A select box, also known as a dropdown list, allows users to choose an option from a list of predefined options.

```
<label
for="country">Country:</label>
<select id="country" name="country"
required>
  <option value="">Select a
country</option>
  <option value="USA">USA</option>
  <option
value="Canada">Canada</option>
  <option value="UK">UK</option>
  <option
value="Australia">Australia</option>
</select>
```

Date

The date input type is used for accepting dates.

```
<label for="date-of-birth">Date of
Birth:</label>
<input type="date" id="date-of-
birth" name="date-of-birth"
required>
```

Time

The time input type is used for accepting time values.

```
<label for="appointment-
time">Appointment Time:</label>
<input type="time" id="appointment-
time" name="appointment-time"
required>
```

FORM VALIDATION

Form validation is the process of checking if the data entered in the form fields meets the specified criteria or not. HTML5 provides a built-in form validation feature that allows you to set rules for the form inputs and validate them easily. Here's an example of how to use form validation in HTML5:

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

```
name="name" required>

  <label for="email">Email:</label>
  <input type="email" id="email"
name="email" required>

  <label
for="password">Password:</label>
  <input type="password"
id="password" name="password"
minlength="8" required>

  <label
for="confirm_password">Confirm
Password:</label>
  <input type="password"
id="confirm_password"
name="confirm_password"
minlength="8" required>

  <input type="submit"
value="Submit">
</form>
```

In the above example, we have a simple form with four input fields: name, email, password, and confirm password. To enable form validation, we have used the `required` attribute for all the fields, which means that the user must fill out all the fields before submitting the form.

For the password and confirm password fields, we have also used the `minlength` attribute to specify the minimum length of the password. If the user enters a password that is shorter than 8 characters, the form will not be submitted, and an error message will be displayed.

When the user submits the form, the browser automatically checks the input fields based on the specified rules and displays error messages if any of the fields are not valid. The error messages are displayed next to the invalid fields, making it easy for the user to correct their mistakes.

Form validation is an essential feature of modern web forms as it helps to ensure that the data entered by the user is correct and valid, which in turn improves the overall user experience.

HTML5 GRAPHICS

HTML5 introduces new elements and attributes for adding graphics to web pages. Here are some of the key features.

CANVAS

The `<canvas>` element allows for dynamic, scriptable rendering of 2D shapes and bitmap images. Here's an example of how to create a canvas element in HTML:

```
<canvas id="myCanvas" width="200"
height="200"></canvas>
```

SVG

The `<svg>` element allows for scalable vector graphics. SVG is a markup language for describing two-dimensional graphics applications and images, and a set of related graphics script interfaces. Here's an example of how to create an SVG element in HTML:

```
<svg width="100" height="100">
  <circle cx="50" cy="50" r="40"
stroke="black" stroke-width="2"
fill="red" />
</svg>
```

AUDIO ELEMENT

The `<audio>` element allows for playing audio on web pages. Here's an example of how to create an audio element in HTML:

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

VIDEO ELEMENT

The <video> element allows for playing videos on web pages. Here's an example of how to create a video element in HTML:

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

CANVAS AND SVG GRAPHICS WITH JAVASCRIPT

Both the <canvas> and <svg> elements can be scripted using JavaScript to create interactive and dynamic graphics. Here's an example of how to draw a circle on a canvas using JavaScript:

```
<canvas id="myCanvas" width="200"
height="200"></canvas>
<script>
  var canvas =
document.getElementById("myCanvas");
  var ctx = canvas.getContext("2d");
  ctx.beginPath();
  ctx.arc(100, 100, 50, 0, 2 *
Math.PI);
  ctx.stroke();
</script>
```

This will draw a circle with a radius of 50 pixels centered at (100, 100) on the canvas.

HTML5 APIS

GEOLOCATION API

This API allows a web application to access the user's location data through the browser.

```
if (navigator.geolocation) {

navigator.geolocation.getCurrentPosi
tion(function(position) {
    console.log("Latitude: " +
position.coords.latitude +
      "Longitude: " +
position.coords.longitude);
  });
} else {
  console.log("Geolocation is not
supported by this browser.");
}
```

WEB STORAGE API

This API allows a web application to store data in the browser's storage, either local or session storage.

```
localStorage.setItem("key",
"value");
console.log(localStorage.getItem("ke
y")); // Outputs "value"
```

CANVAS API

This API allows a web application to dynamically draw graphics and animations on a web page.

```
const canvas =
document.getElementById("myCanvas");
const ctx = canvas.getContext("2d");

ctx.fillStyle = "red";
ctx.fillRect(10, 10, 50, 50);

ctx.strokeStyle = "blue";
ctx.beginPath();
ctx.moveTo(100, 100);
ctx.lineTo(150, 150);
ctx.stroke();
```

WEB AUDIO API

This API allows a web application to play and manipulate audio in real-time, including creating custom audio effects and synthesizers.

```
const context = new AudioContext();
const oscillator =
context.createOscillator();
const gainNode =
context.createGain();

oscillator.connect(gainNode);
gainNode.connect(context.destination
);

oscillator.type = "sawtooth";
oscillator.frequency.value = 440;

gainNode.gain.value = 0.5;

oscillator.start(context.currentTime
);
oscillator.stop(context.currentTime
+ 1);
```

DRAG AND DROP API

The Drag and Drop API in HTML5 allows web developers to make web applications more user-friendly by enabling users to drag and drop elements on a web page. With this API, you can allow users to drag elements such as images, files, or text, and drop them into a designated area on the web page.

The Drag and Drop API consists of a set of events, methods, and properties that enable you to create a drag and drop functionality on your web page.

```
<!DOCTYPE html>
<html>
<head>
    <title>Drag and Drop
Example</title>
    <style>
        #dropzone {
            width: 200px;
            height: 200px;
            border: 2px dashed
black;
            padding: 10px;
            margin: 10px;
        }
        #image {
            width: 100px;
            height: 100px;
            background-color:
orange;
            margin: 10px;
        }
    </style>
</head>
<body>
    Drop Here
    Drag Me

    <script>
        var dragImage =
document.getElementById("image");
        var dropzone =
document.getElementById("dropzone");


dragImage.addEventListener("dragstar
t", function(event) {

event.dataTransfer.setData("text/pla
in", event.target.id);
        });


dropzone.addEventListener("dragover"
, function(event) {
            event.preventDefault();

dropzone.style.backgroundColor =
"yellow";
        });


dropzone.addEventListener("dragleave
", function(event) {

dropzone.style.backgroundColor = "";
        });


dropzone.addEventListener("drop",
function(event) {
            event.preventDefault();
            var data =
event.dataTransfer.getData("text/pla
```

```
in");
            var draggedElement =
document.getElementById(data);

dropzone.appendChild(draggedElement)
;

dropzone.style.backgroundColor = "";
        });
    </script>
</body>
</html>
```

WEB WORKERS API

This API allows a web application to run scripts in the background, freeing up the main thread for other tasks and improving performance.

```
// main.js
const worker = new
Worker("worker.js");

worker.postMessage("Hello from the
main thread!");

worker.onmessage = function(event) {
  console.log("Message received from
worker:", event.data);
};

// worker.js
self.onmessage = function(event) {
  console.log("Message received from
main thread:", event.data);
  self.postMessage("Hello from the
worker thread!");
};
```

BEST PRACTICES

HTML5 is a powerful language that provides a lot of flexibility and functionality to web developers. To ensure that you create efficient and high-performing web pages, it's important to follow some best practices. Here are a few to keep in mind:

- **Use semantic HTML**: Use HTML elements that accurately describe the content they contain. This will make your code more readable and accessible.
- **Keep your code clean**: Use consistent indentation, commenting, and formatting. This will make it easier for you to debug and modify your code later on.
- **Optimize images**: Use image compression tools to reduce the size of your images. Large images can significantly slow down the loading speed of your page.
- **Use external style sheets**: Instead of embedding styles within your HTML code, use an external style sheet. This will make your code cleaner and easier to maintain.
- **Minimize the use of inline styles and scripts**: Using inline styles and scripts can make your code harder to maintain and debug. Instead, use external files for your styles and scripts.
- **Use valid HTML**: Use the W3C validation service to ensure that your HTML code is valid. Valid HTML code will be more consistent across different browsers and devices.
- **Use accessibility features**: Use alt tags for images and provide descriptive text for links. This will ensure that your site is accessible to users with disabilities.

ACCESSIBILITY

Accessibility in web design is about ensuring that everyone, regardless of their physical abilities or limitations, can access and use a website with ease. Here are some best practices for creating accessible websites:

- **Use alt text for images**: Alt text provides a textual alternative to images, which is read by screen readers for visually impaired users. Make sure to describe the image accurately and use concise, descriptive language.
- **Use descriptive link text**: Avoid using generic link text like "click here" or "read more." Instead, use descriptive text that tells users where the link will take them.
- **Ensure proper color contrast**: Use colors with sufficient contrast to make sure that text is readable for users with visual impairments.
- **Use clear, readable fonts**: Use fonts that are

easy to read, and make sure to use appropriate font sizes and spacing.

- **Provide captions and transcripts for multimedia**: Videos should include captions and audio files should have transcripts to ensure that deaf and hard-of-hearing users can access the content.
- **Use ARIA attributes**: Use ARIA (Accessible Rich Internet Applications) attributes to enhance the accessibility of web content for users of assistive technologies.
- **Test with assistive technologies**: Use screen readers, keyboard-only navigation, and other assistive technologies to test your website's accessibility.

SEO

SEO, or Search Engine Optimization, is the practice of optimizing a website's content and structure in order to increase its visibility and ranking on search engine results pages. Here are some best practices for optimizing your HTML5 website for search engines:

- **Use semantic HTML5 tags**: Semantic HTML5 tags like `<header>`, `<nav>`, `<main>`, and `<footer>` can help search engines better understand the content and structure of your website.
- **Include relevant keywords**: Incorporate relevant keywords into your page titles, headings, meta descriptions, and content. But be careful not to overuse keywords or engage in keyword stuffing, which can negatively impact your site's ranking.
- **Use descriptive URLs**: Use descriptive, easy-to-read URLs that include relevant keywords and accurately reflect the content of the page.
- **Optimize images**: Optimize your images by compressing them to reduce their file size and adding descriptive alt tags that accurately describe the image.
- **Provide meta descriptions**: Use descriptive meta descriptions that accurately summarize the content of each page.
- **Use responsive design**: Ensure that your website is optimized for mobile devices by using responsive design techniques.
- **Build high-quality backlinks**: Build high-quality backlinks from reputable websites to help improve your site's visibility and ranking.
- **Monitor and analyze your traffic**: Monitor your website traffic and use analytics tools to gain insights into user behavior and optimize your website accordingly.

PERFORMANCE

HTML5 offers many features to improve website performance, and following best practices can make your pages load faster and perform better. Here are some tips to optimize your website's performance:

- **Minimize HTTP Requests**: Reduce the number of HTTP requests required to load a page. This can be achieved by combining multiple files into a single file or using CSS sprites.
- **Use Cache-Control Headers**: Cache-Control headers allow browsers to cache files locally. By using Cache-Control headers, you can reduce the number of requests that are made to your server.
- **Optimize Images**: Use compressed images to reduce the size of your images. Large image sizes can increase page load times.
- **Use a Content Delivery Network (CDN)**: A CDN can improve your website's performance by caching static content on multiple servers across the world. This helps to reduce the load on your server and speed up page load times.
- **Minimize JavaScript and CSS**: Minimize the amount of JavaScript and CSS required to load a page. This can be achieved by compressing files or by removing unnecessary code.
- **Use Gzip Compression**: Use Gzip compression to compress files before they are sent to the browser. This can reduce the file size of your pages and improve load times.
- **Avoid Redirects**: Redirects add additional HTTP requests and can slow down page load times. Avoid redirects wherever possible.
- **Use Async and Defer Attributes**: Use the async and defer attributes on scripts to speed up page load times. These attributes allow scripts to load asynchronously and do not block the rendering of the page.

HTML5 RESOURCES

Here are some resources for learning more about HTML5:

Resource	Description
HTML5 Boilerplate	A popular front-end template that provides a starting point for building modern, cross-browser compatible web applications
HTML5 UP	A collection of free, responsive HTML5 templates that can be used as a starting point for building websites
CSS-Tricks	A website that provides tutorials, articles, and resources for working with CSS, HTML, and JavaScript
Stack Overflow	A Q&A community for programmers, including those working with HTML5. You can find answers to common questions and ask your own

REFERENCES

Some useful references for HTML5 are:

Reference	Description
W3C HTML5 Specification	The official specification for HTML5 by the World Wide Web Consortium (W3C)
Mozilla Developer Network (MDN)	An extensive resource for HTML5 documentation, tutorials, and examples
HTML5 Rocks	A Google-run site with articles, tutorials, and examples on HTML5 and related technologies
HTML5 Doctor	A site focused on teaching best practices and use cases for HTML5
CanIUse	A website that allows you to check the compatibility of HTML5 features across different web browsers

These references can provide a wealth of information for developers looking to learn or expand their knowledge of HTML5.

TOOLS

There are many tools available for working with HTML5, including:

Tool	Description
Sublime Text	A popular text editor for coding HTML5, with syntax highlighting and autocomplete features
Atom	Another popular text editor with similar features as Sublime Text
Adobe Dreamweaver	Another popular text editor with similar features as Sublime Text
Visual Studio Code	A lightweight code editor with built-in support for HTML5, CSS, and JavaScript
Brackets	An open-source HTML5 editor with live preview and inline editing features
CodePen	An online code editor and community for sharing HTML5, CSS, and JavaScript code snippets

Tool	Description
jsFiddle	An online code editor and playground for HTML5, CSS, and JavaScript, with a focus on front-end development
Gulp	A task runner tool that can automate repetitive tasks in the HTML5 development process, such as compiling SASS files, optimizing images, and concatenating JavaScript files
Grunt	Another task runner tool that can automate similar tasks as Gulp
Adobe Edge Animate	A powerful tool for creating interactive HTML5 animations and multimedia content
Emmet	A productivity tool that can help speed up HTML5 coding by generating code snippets based on simple shorthand syntax

These tools can be useful for developers of all skill levels and can help improve productivity and efficiency when working with HTML5.

FRAMEWORKS AND LIBRARIES

Here are some popular frameworks and libraries used for HTML5 development:

Framework / Library	Description
Bootstrap	A popular CSS framework that allows developers to create responsive websites quickly and easily
Foundation	A responsive front-end framework that provides a set of customizable HTML, CSS, and JavaScript components for building responsive websites
React	A JavaScript library for building user interfaces that can be used to create dynamic and interactive web applications
AngularJS	A popular JavaScript framework for building dynamic web applications that provides a set of tools and features for building single-page applications
Vue.js	A progressive JavaScript framework for building user interfaces that can be used to build web applications
jQuery	A popular JavaScript library that simplifies HTML document manipulation, event handling, and animation
D3.js	A JavaScript library that provides a set of tools for visualizing data using HTML, SVG, and CSS
Three.js	A JavaScript library that provides a set of tools for creating 3D graphics and animations using WebGL
Phaser	A JavaScript library for building HTML5 games that provides a set of tools and features for creating games for desktop and mobile devices

Framework / Library	Description
Ionic	A popular HTML5 framework for building hybrid mobile apps that can be used to build mobile apps using web technologies such as HTML, CSS, and JavaScript