

Learn HTML5 </>

A Beginner's Guide to HTML and Web Graphic

M R KISHORE KUMAR

For you.

One way or another this book ended up in your hands. I'm excited to see what you do with it, and I hope the knowledge within this book makes as large an impact on your life as it has on my own.

About the Tutorial

HTML stands for Hyper Text Markup Language, which is the most widely used language on Web to develop web pages. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2021.

Audience

This tutorial is designed for the aspiring Web Designers and Developers with a need to understand the HTML in enough detail along with its simple overview, and practical examples. This tutorial will give you enough ingredients to start with HTML from where you can take yourself at higher level of expertise.

Prerequisites

Before proceeding with this tutorial you should have a basic working knowledge with Windows or Linux operating system, additionally you must be familiar with:

- Experience with any text editor like notepad, notepad++, or Edit plus etc.
- How to create directories and files on your computer.
- How to navigate through different directories.
- How to type content in a file and save them on a computer.
- Understanding about images in different formats like JPEG, PNG format.

Copyright © 2021 by M R KISHORE KUMAR

All rights reserved. This book or any portion thereof May not be reproduced or used in any manner whatsoever Without the express written permission of the publisher Except for the use of brief quotations in a book review.

Printed in India.

First Printing, 2021

ISBN: 9798599104568

Imprint: Independently Published

www.mrkishorekumar.com

Table of Contents

1) HTML	
Introduction.....	06
2) HTML Editors.....	11
3) HTML Basic	
Examples.....	15
4) HTML Elements.....	18
5) HTML	
Attributes.....	22
6) HTML Headings.....	26
7) HTML	
Paragraphs.....	28
8) HTML Styles.....	31
9) HTML Text	
Formatting.....	33
10) HTML Quotation and Citation Elements.....	37
11) HTML	
Comments.....	41
12) HTML Colors.....	42
13) HTML	
CSS.....	55
14) HTML Links.....	62
15) HTML	
Images.....	69
16) HTML Tables.....	87
17) HTML Lists.....	94
18) HTML Block and Inline Elements.....	105
19) HTML	
Attribute.....	109

20) HTML id	
Attribute.....	114
21) HTML	
Iframes.....	118
22) HTML	
JavaScript.....	121
23) HTML File	
Paths.....	123
24) HTML - The Head	
Element.....	126
25) HTML Layout Elements and	
Techniques.....	133
26) HTML Responsive Web	
Design.....	136
27) HTML Computer Code	
Elements.....	142
28) HTML Semantic	
Elements.....	146
29) HTML Style Guide and Coding Conventions.....	157
30) HTML	
Entities.....	169
31) HTML	
Symbols.....	173
32) Using Emojis in	
HTML.....	176
33) HTML Encoding (Character Sets).....	180
34) HTML Uniform Resource Locators.....	182
35) HTML Versus	
XHTML.....	185

1. HTML Introduction

HTML is the standard markup language for creating Web pages.

What is HTML?

- HTML stands for Hyper Text Markup Language.
- HTML is the standard markup language for creating Web pages.
- HTML describes the structure of a Web page.
- HTML consists of a series of elements.
- HTML elements tell the browser how to display the content.
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

It allows the user to create and structure sections, paragraphs, headings, links, and block quotes for web pages and applications.

HTML is not a programming language, meaning it doesn't have the ability to create dynamic functionality. Instead, it makes it possible to organize and format documents, similarly to Microsoft Word.

A Simple HTML Document

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

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

</body>
</html>
```

Example Explained

- The <!DOCTYPE html> declaration defines that this document is an HTML5 document.
- The <html> element is the root element of an HTML page.
- The <head> element contains meta information about the HTML page.
- The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab).
- The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The <h1> element defines a large heading.
- The <p> element defines a paragraph.

What is an HTML Element?

An HTML element is defined by a start tag, some content, and an end tag:

```
<title>Page Title</title>
```

The HTML **element** is everything from the start tag to the end tag:

```
<h1>My First Heading</h1>
```

```
<p>My first paragraph.</p>
```

Start tag	Element content	End tag
<h1>	My First Heading	</h1>
<p>	My first paragraph	</p>
 	None	None

Note: Some HTML elements have no content (like the
 element). These elements are called empty elements. Empty elements do not have an end tag!

Web Browsers

The purpose of a web browser (Chrome, Edge, Firefox, Safari) is to read HTML documents and display them correctly.

A browser does not display the HTML tags, but uses them to determine how to display the document:



Note : This is the Output of the Simple HTML Document Example.

HTML Page Structure

Below is a visualization of an HTML page structure:



2. HTML Editors

Learn HTML Using Notepad or Text Editor

Web pages can be created and modified by using professional HTML editors.

However, for learning HTML we recommend a simple text editor like Notepad (PC) or Text Editor (Mac).

We believe in that using a simple text editor is a good way to learn HTML.

Follow the steps below to create your first web page with Notepad or Text Editor.

Step 1: Open Notepad (PC)

Windows 8 or later:

Open the **Start Screen** (the window symbol at the bottom left on your screen). Type **Notepad**.

Windows 7 or earlier:

Open **Start > Programs > Accessories > Notepad**

Step 1: Open Text Editor (Mac)

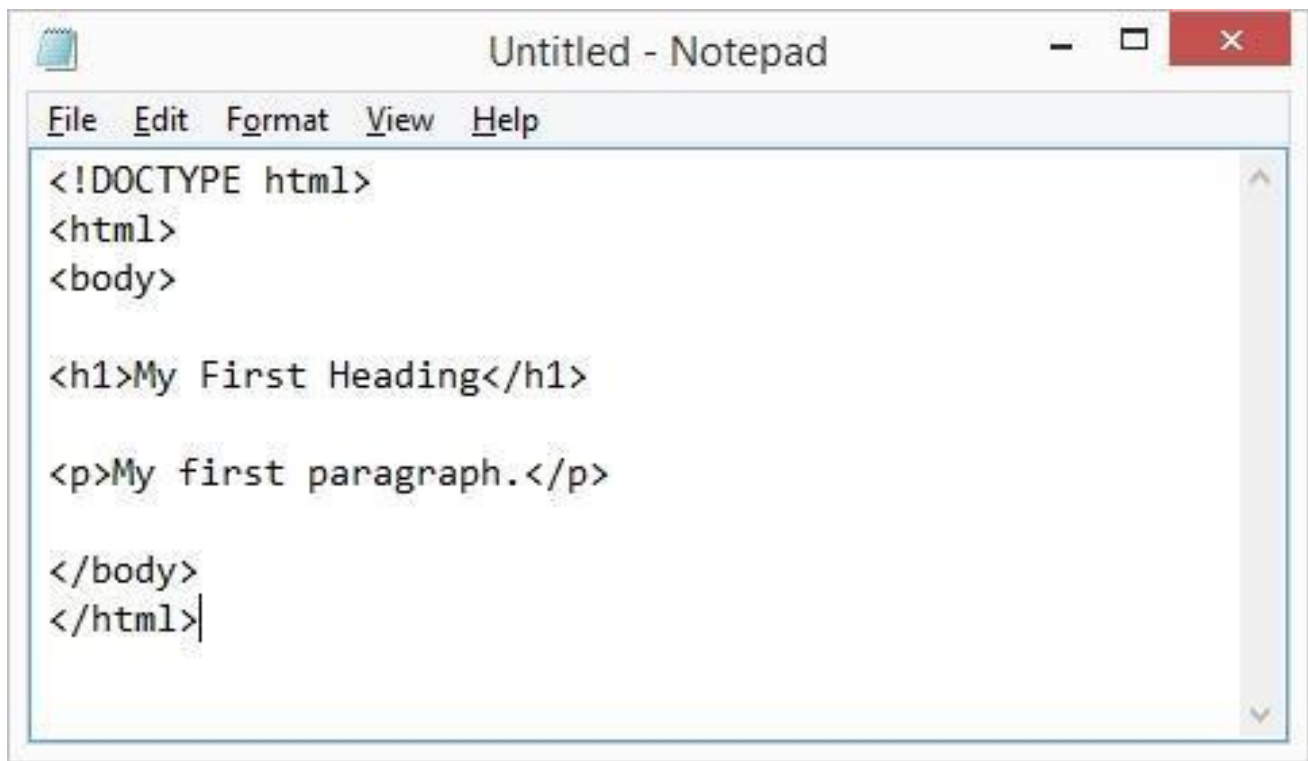
Open **Finder > Applications > Text Editor**

Also change some preferences to get the application to save files correctly. In **Preferences > Format >** choose "**Plain Text**"

Then under "Open and Save", check the box that says "Display HTML files as HTML code instead of formatted text".

Then open a new document to place the code.

Step 2: Write Some HTML

A screenshot of a Notepad application window titled "Untitled - Notepad". The window has a standard Mac OS X title bar with a red close button, a yellow maximize button, and a grey minimize button. Below the title bar is a menu bar with the following items: File, Edit, Format, View, and Help. The main text area contains the following HTML code:

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>

<p>My first paragraph.</p>

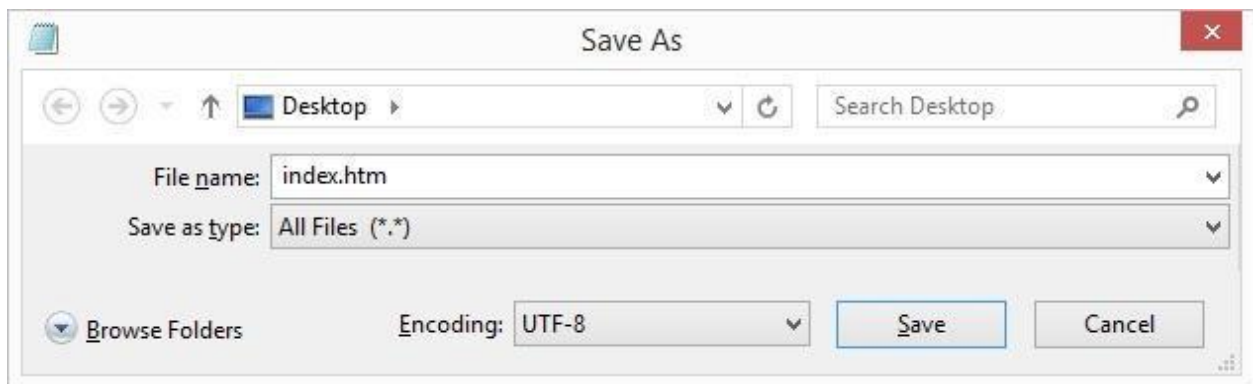
</body>
</html>
```

The cursor is positioned at the end of the last line of code. The text area has a vertical scrollbar on the right side.

Step 3: Save the HTML Page

Save the file on your computer. Select **File > Save as** in the Notepad menu.

Name the file "**index.htm**" and set the encoding to **UTF-8** (which is the preferred encoding for HTML files).



Tip: You can use either .htm or .html as file extension. There is no difference, it is up to you.

Step 4: View the HTML Page in Your Browser

Open the saved HTML file in your favorite browser (double click on the file, or right-click – and choose “Open with”).

The result will look much like this:



Top Free HTML editors

- Atom HTML Editor.
- Sublime Text.
- Adobe Dreamweaver CC.
- Notepad++
- Visual Studio Code.

3. HTML Basic

In this chapter we will show some basic HTML examples.

Don't worry if we use tags you have not learned about yet.

HTML Documents

All HTML documents must start with a document type declaration:

<!DOCTYPE html>.

The HTML document itself begins with **<html>** and ends with **</html>.**

The visible part of the HTML document is between **<body>** and **</body>.**

The <!DOCTYPE> Declaration

The <!DOCTYPE> declaration represents the document type, and helps browsers to display web pages correctly.

It must only appear once, at the top of the page (before any HTML tags).

The <!DOCTYPE> declaration is not case sensitive.

The <!DOCTYPE> declaration for HTML5 is:

```
<!DOCTYPE html>
```

HTML Headings

HTML headings are defined with the <h1> to <h6> tags.

<h1> defines the most important heading. <h6> defines the least important heading:

Example

```
<h1>This is heading 1</h1>  
<h2>This is heading 2</h2>  
<h3>This is heading 3</h3>
```

HTML Paragraphs

HTML paragraphs are defined with the <p> tag:

Example

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

HTML Links

HTML links are defined with the <a> tag:

Example

```
<a href="https://www.mrkishorekumar.com">This is a link</a>
```

The link's destination is specified in the href attribute.

Attributes are used to provide additional information about HTML elements.

You will learn more about attributes in a later chapter.

HTML Images

HTML images are defined with the tag.

The source file (src), alternative text (alt), width, and height are provided as attributes:

Example

```

```

How to View HTML Source?

Have you ever seen a Web page and wondered "Hey! How did they do that?"

View HTML Source Code:

Right-click in an HTML page and select "View Page Source" (in Chrome) or "View Source" (in Edge), or similar in other browsers. This will open a window containing the HTML source code of the page.

Inspect an HTML Element:

Right-click on an element (or a blank area), and choose "Inspect" or "Inspect Element" to see what elements are made up of (you will see both the HTML and the CSS). You can also edit the HTML or CSS on-the-fly in the Elements or Styles panel that opens.

4. HTML Elements

The HTML element is everything from the start tag to the end tag:

```
<title>Page Title</title>
```

Examples of some HTML elements:

```
<h1>My First Heading</h1>
```

```
<p>My first paragraph.</p>
```

Start tag	Element content	End tag
<h1>	My First Heading	</h1>
<p>	My first paragraph	</p>
 	None	None

Note: Some HTML elements have no content (like the
 element). These elements are called empty elements. Empty elements do not have an end tag!

Nested HTML Elements

HTML elements can be nested (this means that elements can contain other elements).

All HTML documents consist of nested HTML elements.

The following example contains four HTML elements (<html>, <body>, <h1> and <p>):

Example

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

Example Explained

The `<html>` element is the root element and it defines the whole HTML document.

It has a start tag `<html>` and an end tag `</html>`.

Then, inside the `<html>` element there is a `<body>` element:

```
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
```

The `<body>` element defines the document's body.

It has a start tag `<body>` and an end tag `</body>`.

Then, inside the `<body>` element there are two other elements: `<h1>` and `<p>`:

`<h1>My First Heading</h1>`

`<p>My first paragraph.</p>`

The `<h1>` element defines a heading.

It has a start tag `<h1>` and an end tag `</h1>`:

`<h1>My First Heading</h1>`

The `<p>` element defines a paragraph.

It has a start tag `<p>` and an end tag `</p>`:

`<p>My first paragraph.</p>`

Never Skip the End Tag

Some HTML elements will display correctly, even if you forget the end tag:

Example

`<html>`

`<body>`

`<p>This is a paragraph`

`<p>This is a paragraph`

`</body>`

`</html>`

However, never rely on this! Unexpected results and errors may occur if you forget the end tag!

Empty HTML Elements

HTML elements with no content are called empty elements.

The `
` tag defines a line break, and is an empty element without a closing tag:

Example

```
<p>This is a <br> paragraph with a line break.</p>
```

HTML is Not Case Sensitive

HTML tags are not case sensitive: `<P>` means the same as `<p>`.

The HTML standard does not require lowercase tags, but it **recommends** lowercase in HTML, and **demand**s lowercase for stricter document types like XHTML.

HTML Tag Reference

Tag reference contains additional information about these tags and their attributes.

Tag	Description
<code><html></code>	Defines the root of an HTML document
<code><body></code>	Defines the document's body
<code><h1></code> to <code><h6></code>	Defines HTML headings

5. HTML Attributes

HTML attributes provide additional information about HTML elements.

HTML Attributes

- All HTML elements can have **attributes**
- Attributes provide **additional information** about elements
- Attributes are always specified in **the start tag**
- Attributes usually come in name/value pairs like: **name="value"**

The href Attribute

The `<a>` tag defines a hyperlink. The **href** attribute specifies the URL of the page the link goes to:

Example

```
<a href="https://www.mrkishorekumar.com">This is a link</a>
```

The src Attribute

The `` tag is used to embed an image in an HTML page. The **src** attribute specifies the path to the image to be displayed:

Example

```

```

There are two ways to specify the URL in the **src** attribute:

1. Absolute URL - Links to an external image that is hosted on another website.

Example: `src="https://www.mrkishorekumar.com/images/img_girl.jpg"`.

Notes: External images might be under copyright. If you do not get permission to use it, you may be in violation of copyright laws. In addition, you cannot control external images; it can suddenly be removed or changed.

2. Relative URL - Links to an image that is hosted within the website. Here, the URL does not include the domain name. If the URL begins without a slash, it will be relative to the current page. Example: `src="img_girl.jpg"`. If the URL begins with a slash, it will be relative to the domain. Example: `src="/images/img_girl.jpg"`.

Tip: It is almost always best to use relative URLs. They will not break if you change domain.

The width and height Attributes

The **** tag should also contain the **width** and **height** attributes, which specifies the width and height of the image (in pixels):

Example

```

```

The alt Attribute

The required **alt** attribute for the **** tag specifies an alternate text for an image, if the image for some reason cannot be displayed. This can be due to slow connection, or an error in the **src** attribute, or if the user uses a screen reader.

```

```

The style Attribute

The **style** attribute is used to add styles to an element, such as color, font, size, and more.

Example

```
<p style="color:red;">This is a red paragraph.</p>
```

The lang Attribute

You should always include the **lang** attribute inside the **<html>** tag, to declare the language of the Web page. This is meant to assist search engines and browsers.

The following example specifies English as the language:

```
<!DOCTYPE html>
<html lang="en">
<body>
...
</body>
</html>
```

Country codes can also be added to the language code in the **lang** attribute. So, the first two characters define the language of the HTML page, and the last two characters define the country.

The following example specifies English as the language and United States as the country:

```
<!DOCTYPE html>
<html lang="en-US">
<body>
...
</body>
</html>
```


The title Attribute

The **title** attribute defines some extra information about an element.

The value of the title attribute will be displayed as a tooltip when you mouse over the element:

Example

```
<p title="I'm a tooltip">This is a paragraph.</p>
```

We Suggest: Always Use Lowercase Attributes

The HTML standard does not require lowercase attribute names.

The title attribute (and all other attributes) can be written with uppercase or lowercase like **title** or **TITLE**.

However, I **recommend** lowercase attributes in HTML, and **demands** lowercase attributes for stricter document types like XHTML.

We Suggest: Always Quote Attribute Values

The HTML standard does not require quotes around attribute values.

However, I **recommend** quotes in HTML, and **demands** quotes for stricter document types like XHTML.

6. HTML Headings

HTML headings are titles or subtitles that you want to display on a webpage.

Example

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

HTML Headings

HTML headings are defined with the `<h1>` to `<h6>` tags.

`<h1>` defines the most important heading. `<h6>` defines the least important heading.

Example

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

Note: Browsers automatically add some white space (a margin) before and after a heading.

Headings Are Important

Search engines use the headings to index the structure and content of your web pages.

Users often skim a page by its headings. It is important to use headings to show the document structure.

<h1> headings should be used for main headings, followed by **<h2>** headings, then the less important **<h3>**, and so on.

Note: Use HTML headings for headings only. Don't use headings to make text BIG or bold.

Bigger Headings

Each HTML heading has a default size. However, you can specify the size for any heading with the **style** attribute, using the CSS **font-size** property:

Example

```
<h1 style="font-size:60px;">Heading 1</h1>
```

7. HTML Paragraphs

A paragraph always starts on a new line, and is usually a block of text.

The HTML `<p>` element defines a paragraph.

A paragraph always starts on a new line, and browsers automatically add some white space (a margin) before and after a paragraph.

Example

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

HTML Display

You cannot be sure how HTML will be displayed.

Large or small screens, and resized windows will create different results.

With HTML, you cannot change the display by adding extra spaces or extra lines in your HTML code.

The browser will automatically remove any extra spaces and lines when the page is displayed:

Example

```
<p>  
This paragraph contains a lot of lines in the source code,  
but the browser ignores it.  
</p>
```

```
<p>  
This paragraph contains a lot of spaces in the source code, but the browser
```

ignores it.

`</p>`

HTML Horizontal Rules

The `<hr>` tag defines a thematic break in an HTML page, and is most often displayed as a horizontal rule.

The `<hr>` element is used to separate content (or define a change) in an HTML page:

Example

```
<h1>This is heading 1</h1>
<p>This is some text.</p>
<hr>
<h2>This is heading 2</h2>
<p>This is some other text.</p>
<hr>
```

The `<hr>` tag is an empty tag, which means that it has no end tag.

HTML Line Breaks

The HTML `
` element defines a line break.

Use `
` if you want a line break (a new line) without starting a new paragraph:

Example

```
<p>This is<br>a paragraph<br>with line breaks.</p>
```

The `
` tag is an empty tag, which means that it has no end tag.

The Poem Problem

This poem will display on a single line:

Example

```
<p>  
  My Bonnie lies over the ocean.  
  
  My Bonnie lies over the sea.  
  
  My Bonnie lies over the ocean.  
  
  Oh, bring back my Bonnie to me.  
</p>
```

Solution - The HTML <pre> Element

The HTML **<pre>** element defines preformatted text.

The text inside a **<pre>** element is displayed in a fixed-width font (usually Courier), and it preserves both spaces and line breaks:

Example

```
<pre>  
  My Bonnie lies over the ocean.  
  
  My Bonnie lies over the sea.  
  
  My Bonnie lies over the ocean.  
  
  Oh, bring back my Bonnie to me.  
</pre>
```

8. HTML Styles

The HTML **style** attribute is used to add styles to an element, such as color, font, size, and more.

Example

I am Red

I am Blue

I am Big

The HTML Style Attribute

Setting the style of an HTML element, can be done with the **style** attribute.

The HTML **style** attribute has the following syntax:

```
<tagname style="property:value;">
```

The **property** is a CSS property. The **value** is a CSS value.

You will learn more about CSS later in this tutorial.

Background Color

The CSS **background-color** property defines the background color for an HTML element.

Example

Set the background color for a page to powderblue:

```
<body style="background-color:powderblue;">  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
  
</body>
```

Set background color for two different elements:

```
<body>  
  
<h1 style="background-color:powderblue;">This is a heading</h1>  
<p style="background-color:tomato;">This is a paragraph.</p>  
  
</body>
```


9. HTML Text Formatting

HTML contains several elements for defining text with a special meaning.

Example :

This text is bold

This text is italic

This is _{subscript} and ^{superscript}

HTML Formatting Elements

Formatting elements were designed to display special types of text:

- **** - Bold text
- **** - Important text
- **<i>** - Italic text
- **** - Emphasized text
- **<mark>** - Marked text
- **<small>** - Smaller text
- **** - Deleted text
- **<ins>** - Inserted text
- **<sub>** - Subscript text
- **<sup>** - Superscript text

HTML `` and `` Elements

The HTML `` element defines bold text, without any extra importance.

Example

``This text is bold``

The HTML `` element defines text with strong importance. The content inside is typically displayed in bold.

Example

``This text is important!``

HTML `<i>` and `` Elements

The HTML `<i>` element defines a part of text in an alternate voice or mood. The content inside is typically displayed in italic.

Tip: The `<i>` tag is often used to indicate a technical term, a phrase from another language, a thought, a ship name, etc.

Example

`<i>`This text is italic`</i>`

The HTML `` element defines emphasized text. The content inside is typically displayed in italic.

Tip: A screen reader will pronounce the words in `` with an emphasis, using verbal stress.

Example

``This text is emphasized``

HTML <small> Element

The HTML <small> element defines smaller text:

Example

```
<small>This is some smaller text.</small>
```

HTML <mark> Element

The HTML <mark> element defines text that should be marked or highlighted:

Example

```
<p>Do not forget to buy <mark>milk</mark> today.</p>
```

HTML Element

The HTML element defines text that has been deleted from a document. Browsers will usually strike a line through deleted text:

Example

```
<p>My favorite color is <del>blue</del> red.</p>
```

HTML <ins> Element

The HTML <ins> element defines a text that has been inserted into a document. Browsers will usually underline inserted text:

Example

```
<p>My favorite color is <del>blue</del> <ins>red</ins>.</p>
```

HTML <sub> Element

The HTML **<sub>** element defines subscript text. Subscript text appears half a character below the normal line, and is sometimes rendered in a smaller font. Subscript text can be used for chemical formulas, like H₂O:

Example

<p>This is **^{**superscripted**}** text.**</p>**

HTML Text Formatting Elements

Tag	Description
	Defines bold text
	Defines emphasized text
<i>	Defines a part of text in an alternate voice or mood
<small>	Defines smaller text
	Defines important text
<sub>	Defines subscripted text
<sup>	Defines superscripted text
<ins>	Defines inserted text
	Defines deleted text
<mark>	Defines marked/highlighted text

10. HTML Quotation and Citation Elements

In this chapter we will go through the `<blockquote>`, `<q>`, `<abbr>`, `<address>`, `<cite>`, and `<bdo>` HTML elements.

HTML `<blockquote>` for Quotations

The HTML `<blockquote>` element defines a section that is quoted from another source.

Browsers usually indent `<blockquote>` elements.

Example

```
<p>Here is a quote from WWF's website:</p>
<blockquote cite="http://www.mrkishorekumar.com/who/index.html">
For 50 years, WWF has been protecting the future of nature.
The world's leading conservation organization,
WWF works in 100 countries and is supported by
1.2 million members in the United States and
close to 5 million globally.
</blockquote>
```

Output

Browsers usually indent blockquote elements.

For nearly 60 years, WWF has been protecting the future of nature. The world's leading conservation organization, WWF works in 100 countries and is supported by more than one million members in the United States and close to five million globally.

HTML <q> for Short Quotations

The HTML **<q>** tag defines a short quotation.

Browsers normally insert quotation marks around the quotation.

Example

```
<p>WWF's goal is to: <q>Build a future where people live in harmony with nature.</q></p>
```

HTML <abbr> for Abbreviations

The HTML **<abbr>** tag defines an abbreviation or an acronym, like "HTML", "CSS", "Mr.", "Dr.", "ASAP", "ATM".

Marking abbreviations can give useful information to browsers, translation systems and search-engines.

Tip: Use the global title attribute to show the description for the abbreviation/acronym when you mouse over the element.

Example

```
<p>The <abbr title="World Health Organization">WHO</abbr> was founded in 1948.</p>
```

HTML <address> for Contact Information

The HTML **<address>** tag defines the contact information for the author/owner of a document or an article.

The contact information can be an email address, URL, physical address, phone number, social media handle, etc.

The text in the `<address>` element usually renders in *italic*, and browsers will always add a line break before and after the `<address>` element.

Example

```
<address>  
Written by John Doe.<br>  
Visit us at:<br>  
Example.com<br>  
Box 564, Disneyland<br>USA  
</address>
```

HTML `<cite>` for Work Title

The HTML `<cite>` tag defines the title of a creative work (e.g. a book, a poem, a song, a movie, a painting, a sculpture, etc.).

Note: A person's name is not the title of a work.

The text in the `<cite>` element usually renders in *italic*.

Example

```
<p><cite>The Scream</cite> by Edvard Munch. Painted in 1893.</p>
```

HTML `<bdo>` for Bi-Directional Override

BDO stands for Bi-Directional Override.

The HTML `<bdo>` tag is used to override the current text direction:

Example

```
<bdo dir="rtl">This text will be written from right to left</bdo>
```

HTML Quotation and Citation Elements

Tag	Description
<abbr>	Defines an abbreviation or acronym
<address>	Defines contact information for the author/owner of a document
<bdo>	Defines the text direction
<blockquote>	Defines a section that is quoted from another source
<cite>	Defines the title of a work
<q>	Defines a short inline quotation

11. HTML Comments

HTML comments are not displayed in the browser, but they can help document your HTML source code.

HTML Comment Tags

You can add comments to your HTML source by using the following syntax:

```
<!-- Write your comments here -->
```

Notice that there is an exclamation point (!) in the start tag, but not in the end tag.

Note: Comments are not displayed by the browser, but they can help document your HTML source code.

Example

```
<!-- This is a comment -->
<p>This is a paragraph.</p>
<!-- Remember to add more information here -->
```

Comments are also great for debugging HTML, because you can comment out HTML lines of code, one at a time, to search for errors:

Example

```
<!-- Do not display this image at the moment

-->
```

12. HTML Colors

HTML colors are specified with predefined color names, or with RGB, HEX, HSL, RGBA, or HSLA values.

Color Names

In HTML, a color can be specified by using a color name:

Tomato
Orange
DodgerBlue
MediumSeaGreen
Gray
SlateBlue
Violet
LightGray

Background Color

You can set the background color for HTML elements:

Hello World

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Example

```
<h1 style="background-color:DodgerBlue;">Hello World</h1>  
<p style="background-color:Tomato;">Lorem ipsum...</p>
```

Text Color

You can set the color of text:

Hello World

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exercitation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Example

```
<h1 style="color:Tomato;">Hello World</h1>  
<p style="color:DodgerBlue;">Lorem ipsum...</p>  
<p style="color:MediumSeaGreen;">Ut wisi enim...</p>
```

Border Color

You can set the color of borders:

Hello World

Hello World

Hello World

Color Values

In HTML, colors can also be specified using RGB values, HEX values, HSL values, RGBA values, and HSLA values.

The following three <div> elements have their background color set with RGB, HEX, and HSL values:

```
rgb(255, 99, 71)
```

```
#ff6347
```

```
hsl(9, 100%, 64%)
```

The following two <div> elements have their background color set with RGBA and HSLA values, which adds an Alpha channel to the color (here we have 50% transparency):

```
rgba(255, 99, 71, 0.5)
```

```
hsla(9, 100%, 64%, 0.5)
```

Example

```
<h1 style="background-color:rgb(255, 99, 71);">...</h1>  
<h1 style="background-color:#ff6347;">...</h1>  
<h1 style="background-color:hsl(9, 100%, 64%);">...</h1>  
  
<h1 style="background-color:rgba(255, 99, 71, 0.5);">...</h1>  
<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">...</h1>
```

Learn more about Color Values

You will learn more about RGB, HEX and HSL in the next chapters.

HTML RGB and RGBA Colors

An RGB color value represents RED, GREEN, and BLUE light sources.

An RGBA color value is an extension of RGB with an Alpha channel (opacity).

RGB Color Values

In HTML, a color can be specified as an RGB value, using this formula:

rgb(red, green, blue)

Each parameter (red, green, and blue) defines the intensity of the color with a value between 0 and 255.

This means that there are $256 \times 256 \times 256 = 16777216$ possible colors!

For example, `rgb(255, 0, 0)` is displayed as red, because red is set to its highest value (255), and the other two (green and blue) are set to 0.

Another example, `rgb(0, 255, 0)` is displayed as green, because green is set to its highest value (255), and the other two (red and blue) are set to 0.

To display black, set all color parameters to 0, like this: `rgb(0, 0, 0)`.

To display white, set all color parameters to 255, like this: `rgb(255, 255, 255)`.

Experiment by mixing the RGB values below:

`rgb(255, 99, 71)`

Example

<code>rgb(255, 0, 0)</code>
<code>rgb(0, 0, 255)</code>
<code>rgb(60, 179, 113)</code>
<code>rgb(238, 130, 238)</code>
<code>rgb(255, 165, 0)</code>
<code>rgb(106, 90, 205)</code>

Shades of Gray

Shades of gray are often defined using equal values for all three parameters:

Example

<code>rgb(60, 60, 60)</code>
<code>rgb(100, 100, 100)</code>
<code>rgb(140, 140, 140)</code>
<code>rgb(180, 180, 180)</code>
<code>rgb(200, 200, 200)</code>
<code>rgb(240, 240, 240)</code>

RGBA Color Values

RGBA color values are an extension of RGB color values with an Alpha channel - which specifies the opacity for a color.

An RGBA color value is specified with:

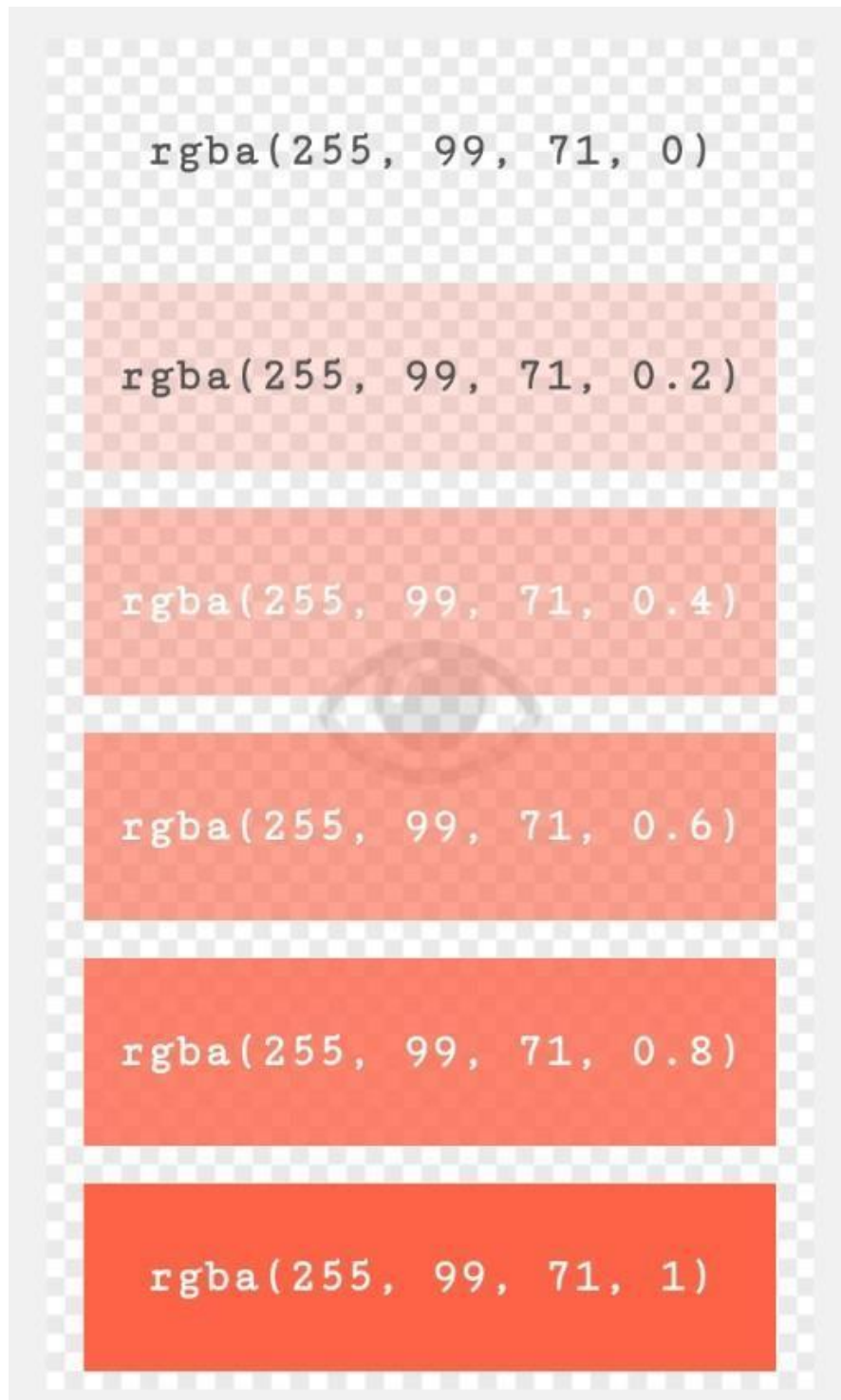
rgba(red, green, blue, alpha)

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):

Experiment by mixing the RGBA values below:

Rgba(255, 99, 71, 0.5)

Example



HTML HEX Colors

A hexadecimal color is specified with: #RRGGBB, where the RR (red), GG (green) and BB (blue) hexadecimal integers specify the components of the color.

HEX Color Values

In HTML, a color can be specified using a hexadecimal value in the form:

#rrggbb

Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).

For example, #ff0000 is displayed as red, because red is set to its highest value (ff), and the other two (green and blue) are set to 00.

Another example, #00ff00 is displayed as green, because green is set to its highest value (ff), and the other two (red and blue) are set to 00.

To display black, set all color parameters to 00, like this: #000000.

To display white, set all color parameters to ff, like this: #ffffff.

Experiment by mixing the HEX values below:

#ff6347

Example

#ff0000
#0000ff
#3cb371
#ee82ee
#ffa500
#6a5acd

Shades of Gray

Shades of gray are often defined using equal values for all three parameters:

Example

#404040
#686868
#a0a0a0
#bebebe
#dcdcdc
#f8f8f8

HTML HSL and HSLA Colors

HSL stands for hue, saturation, and lightness.

HSLA color values are an extension of HSL with an Alpha channel (opacity).

HSL Color Values

In HTML, a color can be specified using hue, saturation, and lightness (HSL) in the form:

hsl(hue, saturation, lightness)

Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue.

Saturation is a percentage value, 0% means a shade of gray, and 100% is the full color.

Lightness is also a percentage value, 0% is black, and 100% is white.

Experiment by mixing the HSL values below:

hsl(0, 100%, 50%)

Example

hsl(0, 100%, 50%)

hsl(240, 100%, 50%)

<code>hsl(147, 50%, 47%)</code>
<code>hsl(300, 76%, 72%)</code>
<code>hsl(39, 100%, 50%)</code>
<code>hsl(248, 53%, 58%)</code>

Saturation

Saturation can be described as the intensity of a color.

100% is pure color, no shades of gray

50% is 50% gray, but you can still see the color.

0% is completely gray, you can no longer see the color.

Example

<code>hsl(0, 100%, 50%)</code>
<code>hsl(0, 80%, 50%)</code>
<code>hsl(0, 60%, 50%)</code>
<code>hsl(0, 40%, 50%)</code>
<code>hsl(0, 20%, 50%)</code>
<code>hsl(0, 0%, 50%)</code>

Lightness

The lightness of a color can be described as how much light you want to give the color, where 0% means no light (black), 50% means 50% light (neither dark nor light) 100% means full lightness (white).

Example

<code>hsl(0, 100%, 0%)</code>
<code>hsl(0, 100%, 25%)</code>
<code>hsl(0, 100%, 50%)</code>
<code>hsl(0, 100%, 75%)</code>
<code>hsl(0, 100%, 90%)</code>
<code>hsl(0, 100%, 100%)</code>

Shades of Gray

Shades of gray are often defined by setting the hue and saturation to 0, and adjust the lightness from 0% to 100% to get darker/lighter shades:

Example

<code>hsl(0, 0%, 20%)</code>
<code>hsl(0, 0%, 30%)</code>
<code>hsl(0, 0%, 40%)</code>
<code>hsl(0, 0%, 60%)</code>
<code>hsl(0, 0%, 70%)</code>
<code>hsl(0, 0%, 90%)</code>

HSLA Color Values

HSLA color values are an extension of HSL color values with an Alpha channel - which specifies the opacity for a color.

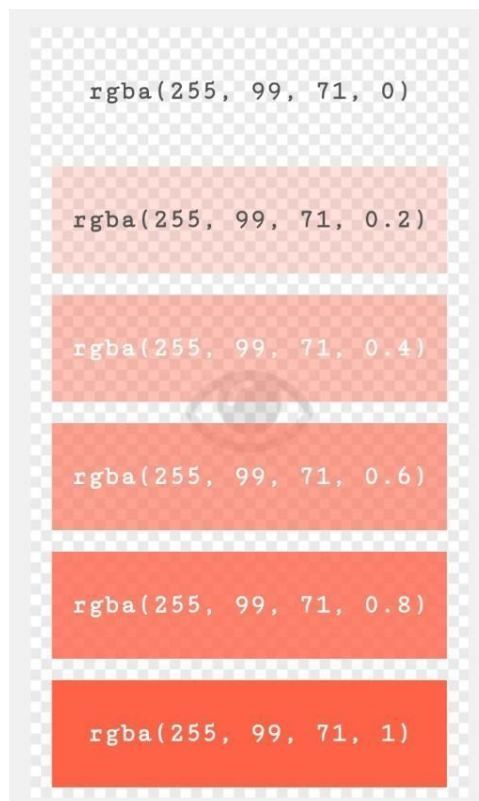
An HSLA color value is specified with:

hsla(hue, saturation, lightness, alpha)

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):

Experiment by mixing the HSLA values below:

Hsla(0, 100%, 50%, 0.5)



13. HTML Styles – CSS

CSS stands for Cascading Style Sheets.

CSS saves a lot of work. It can control the layout of multiple web pages all at once.

CSS = Styles and Colors

Manipulate Text

Colors, Boxes

What is CSS?

Cascading Style Sheets (CSS) is used to format the layout of a webpage.

With CSS, you can control the color, font, the size of text, the spacing between elements, how elements are positioned and laid out, what background images or background colors are to be used, different displays for different devices and screen sizes, and much more!

Tip: The word cascading means that a style applied to a parent element will also apply to all children elements within the parent. So, if you set the color of the body text to “blue”, all headings, paragraphs, and other text elements within the body will also get the same color (unless you specify something else)!

Using CSS

CSS can be added to HTML documents in 3 ways:

- **Inline** - by using the `style` attribute inside HTML elements
- **Internal** - by using a `<style>` element in the `<head>` section
- **External** - by using a `<link>` element to link to an external CSS file

The most common way to add CSS, is to keep the styles in external CSS files. However, in this tutorial we will use inline and internal styles, because this is easier to demonstrate, and easier for you to try it yourself.

Inline CSS

An inline CSS is used to apply a unique style to a single HTML element.

An inline CSS uses the `style` attribute of an HTML element.

The following example sets the text color of the `<h1>` element to blue, and the text color of the `<p>` element to red:

Example

```
<h1 style="color:blue;">A Blue Heading</h1>
```

```
<p style="color:red;">A red paragraph.</p>
```

Internal CSS

An internal CSS is used to define a style for a single HTML page.

An internal CSS is defined in the `<head>` section of an HTML page, within a `<style>` element.

The following example sets the text color of ALL the `<h1>` elements (on that page) to blue, and the text color of ALL the `<p>` elements to red. In addition, the page will be displayed with a "powderblue" background color:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body {background-color: powderblue;}
h1   {color: blue;}
p    {color: red;}
</style>
</head>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
</html>
```

External CSS

An external style sheet is used to define the style for many HTML pages.

To use an external style sheet, add a link to it in the `<head>` section of each HTML page:

Example

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

<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
</html>
```

The external style sheet can be written in any text editor. The file must not contain any HTML code, and must be saved with a .css extension.

Here is what the "styles.css" file looks like:

"styles.css":

```
body {  
  background-color: powderblue;  
}  
h1 {  
  color: blue;  
}  
p {  
  color: red;  
}
```

Tip: With an external style sheet, you can change the look of an entire web site, by changing one file!

CSS Colors, Fonts and Sizes

Here, we will demonstrate some commonly used CSS properties. You will learn more about them later.

The CSS **color** property defines the text color to be used.

The CSS **font-family** property defines the font to be used.

The CSS **font-size** property defines the text size to be used.

Example

Use of CSS color, font-family and font-size properties:

```

<!DOCTYPE html>
<html>
<head>
<style>
h1 {
    color: blue;
    font-family: verdana;
    font-size: 300%;
}
p {
    color: red;
    font-family: courier;
    font-size: 160%;
}
</style>
</head>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
</html>

```

CSS Border

The CSS **border** property defines a border around an HTML element.

Tip: You can define a border for nearly all HTML elements.

Example

Use of CSS border and padding properties:

```

p {
    border: 2px solid powderblue;
    padding: 30px;
}

```

CSS Padding

The CSS **padding** property defines a padding (space) between the text and the border.

Example

Use of CSS border and padding properties:

```
p {  
  border: 2px solid powderblue;  
  padding: 30px;  
}
```

CSS Margin

The CSS **margin** property defines a margin (space) outside the border.

Example

Use of CSS border and margin properties:

```
p {  
  border: 2px solid powderblue;  
  margin: 50px;  
}
```

Link to External CSS

External style sheets can be referenced with a full URL or with a path relative to the current web page.

Example

This example uses a full URL to link to a style sheet:

```
<link rel="stylesheet" href="https://www.sample.com/html/styles.css">
```

Example

This example links to a style sheet located in the html folder on the current web site:

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

Example

This example links to a style sheet located in the same folder as the current page:

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

Chapter Summary

- Use the HTML **style** attribute for inline styling
- Use the HTML **<style>** element to define internal CSS
- Use the HTML **<link>** element to refer to an external CSS file
- Use the HTML **<head>** element to store **<style>** and **<link>** elements
- Use the CSS **color** property for text colors
- Use the CSS **font-family** property for text fonts
- Use the CSS **font-size** property for text sizes
- Use the CSS **border** property for borders
- Use the CSS **padding** property for space inside the border
- Use the CSS **margin** property for space outside the border

HTML Style Tags

<style> Defines style information for an HTML document.

<link> Defines a link between a document and an external resource.

14. HTML Links

Links are found in nearly all web pages. Links allow users to click their way from page to page.

HTML Links – Hyperlinks

HTML links are hyperlinks.

You can click on a link and jump to another document.

When you move the mouse over a link, the mouse arrow will turn into a little hand.

Note: A link does not have to be text. A link can be an image or any other HTML element!

HTML Links – Syntax

The HTML **<a>** tag defines a hyperlink. It has the following syntax:

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

The most important attribute of the **<a>** element is the **href** attribute, which indicates the link's destination.

The **link text** is the part that will be visible to the reader.

Clicking on the link text, will send the reader to the specified URL address.

Example

This example shows how to create a link to mrkishorekumar.com:

```
<a href="https://www.mrkishorekumar.com/">Visit mrkishorekumar.com!</a>
```

By default, links will appear as follows in all browsers:

- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red

Tip: Links can of course be styled with CSS, to get another look!

HTML Links – The target Attribute

By default, the linked page will be displayed in the current browser window. To change this, you must specify another target for the link.

The **target** attribute specifies where to open the linked document.

The **target** attribute can have one of the following values:

- **_self** - Default. Opens the document in the same window/tab as it was clicked
- **_blank** - Opens the document in a new window or tab
- **_parent** - Opens the document in the parent frame
- **_top** - Opens the document in the full body of the window

Example

Use target="_blank" to open the linked document in a new browser window or tab:

```
<a href="https://www.mrkishorekumar.com/" target="_blank">Visit  
mrkishorekumar!</a>
```

Absolute URLs vs. Relative URLs

Both examples above are using an **absolute URL** (a full web address) in the **href** attribute.

A local link (a link to a page within the same website) is specified with a **relative URL** (without the "https://www" part):

Example

```
<h2>Absolute URLs</h2>
<p><a href="https://www.youtube.com/">YouTube</a></p>
<p><a href="https://www.google.com/">Google</a></p>

<h2>Relative URLs</h2>
<p><a href="html_images.asp">HTML Images</a></p>
<p><a href="/css/default.asp">CSS Tutorial</a></p>
```

HTML Links – Use an Image as a Link

To use an image as a link, just put the **img** tag inside the **a** tag:

Example

```
<a href="default.asp">

</a>
```

Link to an Email Address

Use **mailto:** inside the **href** attribute to create a link that opens the user's email program (to let them send a new email):

```
<a href="mailto:someone@example.com">Send email</a>
```


Button as a Link

To use an HTML button as a link, you have to add some JavaScript code. JavaScript allows you to specify what happens at certain events, such as a click of a button:

Example

```
<button onclick="document.location='default.asp'">HTML Tutorial</button>
```

Link Titles

The **title** attribute specifies extra information about an element. The information is most often shown as a tooltip text when the mouse moves over the element.

Example

```
<a href="https://www.Mrkishorekumar.com/html/" title="Go to MRKK-HTML section">Visit our HTML Tutorial</a>
```

More on Absolute URLs and Relative URLs

Examples

Use a full URL to link to a web page:

```
<a href="https://www.sample.com/html/default.asp">HTML tutorial</a>
```

Link to a page located in the html folder on the current web site:

```
<a href="/html/default.asp">HTML tutorial</a>
```

Link to a page located in the same folder as the current page:

```
<a href="default.asp">HTML tutorial</a>
```

HTML Links – Different Colors

An HTML link is displayed in a different color depending on whether it has been visited, is unvisited, or is active.

HTML Link Colors

By default, a link will appear like this (in all browsers):

- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red

You can change the link state colors, by using CSS:

Example

Here, an unvisited link will be green with no underline. A visited link will be pink with no underline. An active link will be yellow and underlined. In addition, when mousing over a link (a:hover) it will become red and underlined:

```
<style>
a:link {
  color: green;
  background-color: transparent;
  text-decoration: none;
}

a:visited {
  color: pink;
  background-color: transparent;
  text-decoration: none;
}

a:hover {
  color: red;
  background-color: transparent;
  text-decoration: underline;
}
```

```
a:active {  
  color: yellow;  
  background-color: transparent;  
  text-decoration: underline;  
}  
</style>
```

Link Buttons

A link can also be styled as a button, by using CSS:

This is a link

Example

```
<style>  
a:link, a:visited {  
  background-color: #f44336;  
  color: white;  
  padding: 15px 25px;  
  text-align: center;  
  text-decoration: none;  
  display: inline-block;  
}  
  
a:hover, a:active {  
  background-color: red;  
}  
</style>
```

HTML Links – Create Bookmarks

HTML links can be used to create bookmarks, so that readers can jump to specific parts of a web page.

Create a Bookmark in HTML

Bookmarks can be useful if a web page is very long.

To create a bookmark - first create the bookmark, then add a link to it.

When the link is clicked, the page will scroll down or up to the location with the bookmark.

Example

First, use the **id** attribute to create a bookmark:

```
<h2 id="C4">Chapter 4</h2>
```

Then, add a link to the bookmark ("Jump to Chapter 4"), from within the same page:

Example

```
<a href="#C4">Jump to Chapter 4</a>
```

You can also add a link to a bookmark on another page:

```
<a href="html_demo.html#C4">Jump to Chapter 4</a>
```

Chapter Summary

- Use the **id** attribute (`id="value"`) to define bookmarks in a page
- Use the **href** attribute (`href="#value"`) to link to the bookmark

15. HTML Images

Images can improve the design and the appearance of a web page.



Example

```

```

HTML Images Syntax

The HTML `` tag is used to embed an image in a web page.

Images are not technically inserted into a web page; images are linked to web pages. The `` tag creates a holding space for the referenced image.

The `` tag is empty, it contains attributes only, and does not have a closing tag.

The `` tag has two required attributes:

- `src` - Specifies the path to the image

- alt - Specifies an alternate text for the image

Syntax

```

```

The src Attribute

The required **src** attribute specifies the path (URL) to the image.

Note: When a web page loads; it is the browser, at that moment, that gets the image from a web server and inserts it into the page. Therefore, make sure that the image actually stays in the same spot in relation to the web page, otherwise your visitors will get a broken link icon. The broken link icon and the **alt** text are shown if the browser cannot find the image.

Example

```

```

The alt Attribute

The required **alt** attribute provides an alternate text for an image, if the user for some reason cannot view it (because of slow connection, an error in the src attribute, or if the user uses a screen reader).

The value of the **alt** attribute should describe the image:

Example

```

```

If a browser cannot find an image, it will display the value of the **alt** attribute:

Example

```

```

Tip: A screen reader is a software program that reads the HTML code, and allows the user to “listen” to the content. Screen readers are useful for people who are visually impaired or learning disabled.

Image Size – Width and Height

You can use the **style** attribute to specify the width and height of an image.

Example

```

```

Alternatively, you can use the **width** and **height** attributes:

Example

```

```

The **width** and **height** attributes always define the width and height of the image in pixels.

Note: Always specify the width and height of an image. If width and height are not specified, the web page might flicker while the image loads.

Width and Height, or Style?

The **width**, **height**, and **style** attributes are all valid in HTML.

However, we suggest using the **style** attribute. It prevents styles sheets from changing the size of images:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  width: 100%;
}
</style>
</head>
<body>





</body>
</html>
```

Images in Another Folder

If you have your images in a sub-folder, you must include the folder name in the **src** attribute:

Example

```

```

Images on Another Server/Website

Some web sites points to an external image on another server.

To point to an image on another server, you must specify an absolute (full) URL in the **src** attribute:

```

```


Notes on external images: External images might be under copyright. If you do not get permission to use it, you may be in violation of copyright laws. In addition, you cannot control external images; it can suddenly be removed or changed.

Animated Images

HTML allows animated GIFs:

Example

```

```

Image as a Link

To use an image as a link, put the **** tag inside the **<a>** tag:

Example

```
<a href="default.asp">  
    
</a>
```

Image Floating

Use the CSS **float** property to let the image float to the right or to the left of a text:

Example

```
<p>  
The image will float to the right of the text.</p>  
  
<p>  
The image will float to the left of the text.</p>

## Common Image Formats

Here are the most common image file types, which are supported in all browsers (Chrome, Edge, Firefox, Safari, Opera):

| Abbreviation | File Format                           | File Extension                   |
|--------------|---------------------------------------|----------------------------------|
| APNG         | Animated Portable Network Graphics    | .apng                            |
| GIF          | Graphics Interchange Format           | .gif                             |
| ICO          | Microsoft Icon                        | .ico, .cur                       |
| JPEG         | Joint Photographic Expert Group image | .jpg, .jpeg, .jfif, .pjpeg, .jpp |
| PNG          | Portable Network Graphics             | .png                             |
| SVG          | Scalable Vector Graphics              | .svg                             |

## Chapter Summary

- Use the HTML `<img>` element to define an image
- Use the HTML `src` attribute to define the URL of the image
- Use the HTML `alt` attribute to define an alternate text for an image, if it cannot be displayed
- Use the HTML `width` and `height` attributes or the CSS `width` and `height` properties to define the size of the image
- Use the CSS `float` property to let the image float to the left or to the right

# HTML Image Maps

With HTML image maps, you can create clickable areas on an image.

## Image Maps

The HTML `<map>` tag defines an image map. An image map is an image with clickable areas. The areas are defined with one or more `<area>` tags.

Try to click on the computer, phone, or the cup of coffee in the image below:



## Example

Here is the HTML source code for the image map above:

```


<map name="workmap">
 <area shape="rect" coords="34,44,270,350" alt="Computer" href="computer.
htm">
 <area shape="rect" coords="290,172,333,250" alt="Phone" href="phone.htm"
>
 <area shape="circle" coords="337,300,44" alt="Coffee" href="coffee.htm">
</map>
```

## How Does it Work?

The idea behind an image map is that you should be able to perform different actions depending on where in the image you click.

To create an image map you need an image, and some HTML code that describes the clickable areas.

## The Image

The image is inserted using the `<img>` tag. The only difference from other images is that you must add a `usemap` attribute:

```

```

The `usemap` value starts with a hash tag `#` followed by the name of the image map, and is used to create a relationship between the image and the image map.

## Create Image Map

Then, add a `<map>` element.

The `<map>` element is used to create an image map, and is linked to the image by using the required `name` attribute:

```
<map name="workmap">
```

The `name` attribute must have the same value as the `<img>`'s `usemap` attribute .

## The Areas

Then, add the clickable areas.

A clickable area is defined using an **<area>** element.

## Shape

You must define the shape of the clickable area, and you can choose one of these values:

- **rect** - defines a rectangular region
- **circle** - defines a circular region
- **poly** - defines a polygonal region
- **default** - defines the entire region

You must also define some coordinates to be able to place the clickable area onto the image.

### Shape="rect"

The coordinates for **shape="rect"** come in pairs, one for the x-axis and one for the y-axis.

So, the coordinates **34,44** is located 34 pixels from the left margin and 44 pixels from the top:



The coordinates **270,350** is located 270 pixels from the left margin and 350 pixels from the top:



Now we have enough data to create a clickable rectangular area:



## Example

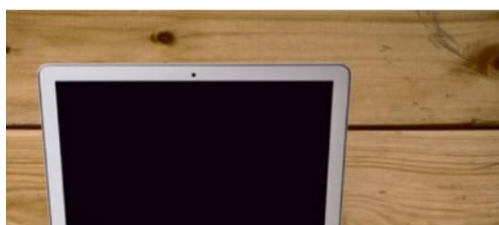
```
<area shape="rect" coords="34, 44, 270, 350" href="computer.htm">
```

**Shape="circle"** To add a circle area, first locate the coordinates of the center of the circle: **337,300**



Then specify the radius of the circle:

**44** pixels





**Now you have enough data to create a clickable circular area:**

## Example

```
<area shape="circle" coords="337, 300, 44" href="coffee.htm">
```

This is the area that becomes clickable and will send the user to the page “coffee.htm”:

## Shape="poly"

The **shape="poly"** contains several coordinate points, which creates a shape formed with straight lines (a polygon).

This can be used to create any shape.

Like maybe a croissant shape!

How can we make the croissant in the image below become a clickable link?

We have to find the x and y coordinates for all edges of the croissant:

The coordinates come in pairs, one for the x-axis and one for the y-axis:

## Example

```
<area shape="poly" coords="140,121,181,116,204,160,204,222,191,270,140,329,85,355,58,352,37,322,40,259,103,161,128,147" href="croissant.htm">
```

This is the area that becomes clickable and will send the user to the page “croissant.htm”:

## Image Map and JavaScript

A clickable area can also trigger a JavaScript function.

Add a click event to the <area> element to execute a JavaScript function:

## Example

Here, we use the onclick attribute to execute a JavaScript function when the area is clicked:

```
<map name="workmap">
 <area shape="circle" coords="337,300,44" onclick="myFunction()">
</map>

<script>
function myFunction() {
 alert("You clicked the coffee cup!");
}
</script>
```



## Chapter Summary

- Use the HTML `<map>` element to define an image map
- Use the HTML `<area>` element to define the clickable areas in the image map
- Use the HTML `usemap` attribute of the `<img>` element to point to an image map

## HTML Background Images

A background image can be specified for almost any HTML element.

### Background Image on a HTML element

To add a background image on an HTML element, use the HTML `style` attribute and the CSS `background-image` property:

#### Example

Add a background image on a HTML element:

```
<div style="background-image: url('img_girl.jpg');">
```

You can also specify the background image in the `<style>` element, in the `<head>` section:

#### Example

Specify the background image in the `<style>` element:

```
<style>
div {
 background-image: url('img_girl.jpg');
}
</style>
```

## Background Image on a Page

If you want the entire page to have a background image, you must specify the background image on the `<body>` element:

### Example

Add a background image for the entire page:

```
<style>
body {
 background-image: url('img_girl.jpg');
}
</style>
```

## Background Repeat

If the background image is smaller than the element, the image will repeat itself, horizontally and vertically, until it reaches the end of the element.

### Example

```
<style>
body {
 background-image: url('example_img_girl.jpg');
}
</style>
```

To avoid the background image from repeating itself, set the `background-repeat` property to `no-repeat`.

### Example

```
<style>
body {
 background-image: url('example_img_girl.jpg');
 background-repeat: no-repeat;
}
</style>
```

## Background Cover

If you want the background image to cover the entire element, you can set the **background-size** property to **cover**.

Also, to make sure the entire element is always covered, set the **background-attachment** property to **fixed**:

This way, the background image will cover the entire element, with no stretching (the image will keep its original proportions):

### Example

```
<style>
body {
 background-image: url('img_girl.jpg');
 background-repeat: no-repeat;
 background-attachment: fixed;
 background-size: cover;
}
</style>
```

## Background Stretch

If you want the background image to stretch to fit the entire element, you can set the **background-size** property to **100% 100%**:

Try resizing the browser window, and you will see that the image will stretch, but always cover the entire element.

### Example

```
<style>
body {
 background-image: url('img_girl.jpg');
 background-repeat: no-repeat;
 background-attachment: fixed;
 background-size: 100% 100%;
}
</style>
```

## Learn More CSS

From the examples above you have learned that background images can be styled by using the CSS background properties.

## HTML <picture> Element

The HTML **<picture>** element allows you to display different pictures for different devices or screen sizes.

### The HTML <picture> Element

The HTML **<picture>** element gives web developers more flexibility in specifying image resources.

The **<picture>** element contains one or more **<source>** elements, each referring to different images through the **srcset** attribute. This way the browser can choose the image that best fits the current view and/or device.

Each **<source>** element has a **media** attribute that defines when the image is the most suitable.

### Example

Show different images for different screen sizes:

```
<picture>
 <source media="(min-width: 650px)" srcset="img_food.jpg">
 <source media="(min-width: 465px)" srcset="img_car.jpg">

</picture>
```

**Note:** Always specify an <img> element as the last child element of the <picture> element. The <img> element is used by browsers that do not support the <picture> element, or if none of the <source> tags match.

## When to use the Picture Element

There are two main purposes for the **<picture>** element:

### 1. Bandwidth

If you have a small screen or device, it is not necessary to load a large image file. The browser will use the first **<source>** element with matching attribute values, and ignore any of the following elements.

### 2. Format Support

Some browsers or devices may not support all image formats. By using the **<picture>** element, you can add images of all formats, and the browser will use the first format it recognizes, and ignore any of the following elements.

## Example

The browser will use the first image format it recognizes:

```
<picture>
 <source srcset="img_avatar.png">
 <source srcset="img_girl.jpg">

</picture>
```

**Note:** The browser will use the first <source> element with matching attribute values, and ignore any following <source> elements.

## HTML Image Tags

Tag	Description
<img>	Defines an image
<map>	Defines an image map
<area>	Defines a clickable area inside an image map
<picture>	Defines a container for multiple image resources

## 16. HTML Tables

HTML tables allow web developers to arrange data into rows and columns.

### Example

Company	Contact	Country
Alfreds Futterkiste	Maria Anders	Germany
Centro comercial Moctezuma	Francisco Chang	Mexico
Ernst Handel	Roland Mendel	Austria
Island Trading	Helen Bennett	UK
Laughing Bacchus Winecellars	Yoshi Tannamuri	Canada
Magazzini Alimentari Riuniti	Giovanni Rovelli	Italy

### Define an HTML Table

The **<table>** tag defines an HTML table.

Each table row is defined with a **<tr>** tag. Each table header is defined with a **<th>** tag. Each table data/cell is defined with a **<td>** tag.

By default, the text in **<th>** elements are bold and centered.

By default, the text in **<td>** elements are regular and left-aligned.

## Example

A simple HTML table:

```
<table style="width:100%">
 <tr>
 <th>First name</th>
 <th>Last name</th>
 <th>Age</th>
 </tr>
 <tr>
 <td>Jill</td>
 <td>Smith</td>
 <td>50</td>
 </tr>
 <tr>
 <td>Eve</td>
 <td>Jackson</td>
 <td>94</td>
 </tr>
</table>
```

**Note:** The <td> elements are the data containers of the table.

They can contain all sorts of HTML elements; text, images, lists, other tables, etc.

## HTML Table – Add a Border

To add a border to a table, use the CSS **border** property:

## Example

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

Remember to define borders for both the table and the table cells.



## HTML Table – Collapsed Borders

To let the borders collapse into one border, add the CSS **border-collapse** property:

### Example

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

## HTML Table – Add Cell Padding

Cell padding specifies the space between the cell content and its borders.

If you do not specify a padding, the table cells will be displayed without padding.

To set the padding, use the CSS **padding** property:

### Example

```
th, td {
 padding: 15px;
}
```

## HTML Table – Left-align Headings

By default, table headings are bold and centered.

To left-align the table headings, use the CSS **text-align** property:

## Example

```
th {
 text-align: left;
}
```

## HTML Table – Add Border Spacing

Border spacing specifies the space between the cells.

To set the border spacing for a table, use the CSS **border-spacing** property:

## Example

```
table {
 border-spacing: 5px;
}
```

**Note:** If the table has collapsed borders, border-spacing has no effect.

## Example

```
<table style="width:100%">
 <tr>
 <th>Name</th>
 <th colspan="2">Telephone</th>
 </tr>
 <tr>
 <td>Bill Gates</td>
 <td>55577854</td>
 <td>55577855</td></tr>
</table>
```

## HTML Table – Cell that Spans Many Rows

To make a cell span more than one row, use the **rowspan** attribute:

## Example

```
<table style="width:100%">
 <tr>
 <th>Name:</th>
 <td>Bill Gates</td>
 </tr>
 <tr>
 <th rowspan="2">Telephone:</th>
 <td>55577854</td>
 </tr>
 <tr>
 <td>55577855</td>
 </tr>
</table>
```

## HTML Table – Add a Caption

To add a caption to a table, use the **<caption>** tag:

```
<table style="width:100%">
 <caption>Monthly savings</caption>
 <tr>
 <th>Month</th>
 <th>Savings</th>
 </tr>
 <tr>
 <td>January</td>
 <td>$100</td>
 </tr>
 <tr>
 <td>February</td>
 <td>$50</td>
 </tr>
</table>
```

**Note:** The <caption> tag must be inserted immediately after the <table> tag.

```
<table id="t01">
 <tr>
 <th>First name</th>
 <th>Last name</th>
 <th>Age</th>
 </tr>
 <tr>
 <td>Eve</td>
 <td>Jackson</td>
 <td>94</td>
 </tr>
</table>
```

Now you can define a special style for this table:

```
#t01 {
 width: 100%;
 background-color: #f1f1c1;
}
```

And add more styles:

```
#t01 tr:nth-child(even) {
 background-color: #eee;
}
#t01 tr:nth-child(odd) {
 background-color: #fff;
}
#t01 th {
 color: white;
 background-color: black;
}
```

## Chapter Summary

- Use the HTML `<table>` element to define a table
- Use the HTML `<tr>` element to define a table row
- Use the HTML `<td>` element to define a table data

- Use the HTML `<th>` element to define a table heading
- Use the HTML `<caption>` element to define a table caption
- Use the CSS `border` property to define a border
- Use the CSS `border-collapse` property to collapse cell borders
- Use the CSS `padding` property to add padding to cells
- Use the CSS `text-align` property to align cell text
- Use the CSS `border-spacing` property to set the spacing between cells
- Use the `colspan` attribute to make a cell span many columns
- Use the `rowspan` attribute to make a cell span many rows
- Use the `id` attribute to uniquely define one table

## 17. HTML Lists

HTML lists allow web developers to group a set of related items in lists.

### Example

An unordered HTML list:

- Item
- Item
- Item
- Item

An ordered HTML list:

1. First item
2. Second item
3. Third item
4. Fourth item

### Unordered HTML List

An unordered list starts with the `<ul>` tag. Each list item starts with the `<li>` tag.

The list items will be marked with bullets (small black circles) by default:

## Example

```

 Coffee
 Tea
 Milk

```

## Ordered HTML List

An ordered list starts with the **<ol>** tag. Each list item starts with the **<li>** tag.

The list items will be marked with numbers by default:

## Example

```

 Coffee
 Tea
 Milk

```

## HTML Description Lists

HTML also supports description lists.

A description list is a list of terms, with a description of each term.

The **<dl>** tag defines the description list, the **<dt>** tag defines the term (name), and the **<dd>** tag describes each term:

## Example

```
<dl>
 <dt>Coffee</dt>
 <dd>- black hot drink</dd>
 <dt>Milk</dt>
 <dd>- white cold drink</dd>
</dl>
```

## HTML List Tags

Tag	Description
<ul>	Defines an unordered list
<ol>	Defines an ordered list
<li>	Defines a list item
<dl>	Defines a description list
<dt>	Defines a term in a description list
<dd>	Describes the term in a description list

## HTML Unordered Lists

The HTML `<ul>` tag defines an unordered (bulleted) list.

### Unordered HTML List

An unordered list starts with the `<ul>` tag. Each list item starts with the `<li>` tag.

The list items will be marked with bullets (small black circles) by default:

### Example

```

 Coffee
 Tea
```



```
Milk

```

## Unordered HTML List - Choose List Item Marker

The CSS `list-style-type` property is used to define the style of the list item marker. It can have one of the following values:

Value	Description
disc	Sets the list item marker to a bullet (default)
circle	Sets the list item marker to a circle
square	Sets the list item marker to a square
none	The list items will not be marked

### Example - Disc

```
<ul style="list-style-type:disc;">
 Coffee
 Tea
 Milk

```

### Example - Circle

```
<ul style="list-style-type:circle;">
 Coffee
 Tea
 Milk

```

### Example - None

```
<ul style="list-style-type:none;">
 Coffee
 Tea
 Milk

```

## Nested HTML Lists

Lists can be nested (list inside list):

### Example

```

 Coffee
 Tea

 Black tea
 Green tea

 Milk

```

**Note:** A list item (<li>) can contain a new list, and other HTML elements, like images and links, etc.

## Horizontal List with CSS

HTML lists can be styled in many different ways with CSS.

One popular way is to style a list horizontally, to create a navigation menu:

### Example

```
<!DOCTYPE html>
<html>
<head>
<style>
ul {
 list-style-type: none;
```

```

 margin: 0;
 padding: 0;
 overflow: hidden;
 background-color: #333333;
}

li {
 float: left;
}

li a {
 display: block;
 color: white;
 text-align: center;
 padding: 16px;
 text-decoration: none;
}

li a:hover {
 background-color: #111111;
}
</style>
</head>
<body>

 Home
 News
 Contact
 About

</body>
</html>

```

## Chapter Summary

- Use the HTML `<ul>` element to define an unordered list
- Use the CSS `list-style-type` property to define the list item marker
- Use the HTML `<li>` element to define a list item

- Lists can be nested
- List items can contain other HTML elements
- Use the CSS property `float:left` to display a list horizontally

## HTML Ordered Lists

The HTML `<ol>` tag defines an ordered list. An ordered list can be numerical or alphabetical.

### Ordered HTML List

An ordered list starts with the `<ol>` tag. Each list item starts with the `<li>` tag.

The list items will be marked with numbers by default:

### Example

```

 Coffee
 Tea
 Milk

```

## Ordered HTML List – The Type Attribute

The **type** attribute of the `<ol>` tag, defines the type of the list item marker:

Type	Description
type="1"	The list items will be numbered with numbers (default)
type="A"	The list items will be numbered with uppercase letters
type="a"	The list items will be numbered with lowercase letters
type="I"	The list items will be numbered with uppercase roman numbers
type="i"	The list items will be numbered with lowercase roman numbers

## Numbers:

```
<ol type="1">
 Coffee
 Tea
 Milk

```

## Uppercase Letters:

```
<ol type="A">
 Coffee
 Tea
 Milk

```

## Lowercase Letters:

```
<ol type="a">
 Coffee
 Tea
 Milk

```

## Uppercase Roman Numbers:

```
<ol type="I">
 Coffee
 Tea
 Milk

```

## Lowercase Roman Numbers:

```
<ol type="i">
 Coffee
 Tea
 Milk

```

## Control List Counting

By default, an ordered list will start counting from 1. If you want to start counting from a specified number, you can use the **start** attribute:

### Example

```
<ol start="50">
 Coffee
 Tea
 Milk

```

**Nested HTML Lists** – Lists can be nested (list inside list):

### Example

```

 Coffee
 Tea

 Black tea
 Green tea

 Milk

```

**Note:** A list item (<li>) can contain a new list, and other HTML elements, like images and links, etc.

## Chapter Summary

- Use the HTML **<ol>** element to define an ordered list
- Use the HTML **type** attribute to define the numbering type
- Use the HTML **<li>** element to define a list item
- Lists can be nested
- List items can contain other HTML elements

# HTML Other Lists

HTML also supports description lists.

## HTML Description Lists

A description list is a list of terms, with a description of each term.

The `<dl>` tag defines the description list, the `<dt>` tag defines the term (name), and the `<dd>` tag describes each term:

### Example

```
<dl>
 <dt>Coffee</dt>
 <dd>- black hot drink</dd>
 <dt>Milk</dt>
 <dd>- white cold drink</dd>
</dl>
```

## Chapter Summary

- Use the HTML `<dl>` element to define a description list
- Use the HTML `<dt>` element to define the description term
- Use the HTML `<dd>` element to describe the term in a description list



## 18. HTML Block and Inline Elements

Every HTML element has a default display value, depending on what type of element it is.

There are two display values: block and inline.

### Block-level Elements

A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).

**The <div> element is a block-level element.**

#### Example

```
<div>Hello World</div>
```

### Inline Elements

An inline element does not start on a new line and it only takes up as much width as necessary.

This is a **<span> element inside** a paragraph.

#### Example

```
Hello World
```

**Here are the block-level elements in HTML**

<code>&lt;address&gt;</code>	<code>&lt;article&gt;</code>
<code>&lt;aside&gt;</code>	<code>&lt;blockquote&gt;</code>
<code>&lt;canvas&gt;</code>	<code>&lt;dd&gt;</code>
<code>&lt;div&gt;</code>	<code>&lt;dl&gt;</code>
<code>&lt;dt&gt;</code>	<code>&lt;fieldset&gt;</code>
<code>&lt;figcaption&gt;</code>	<code>&lt;figure&gt;</code>
<code>&lt;footer&gt;</code>	<code>&lt;form&gt;</code>
<code>&lt;h1&gt;-&lt;h6&gt;</code>	<code>&lt;header&gt;</code>
<code>&lt;hr&gt;</code>	<code>&lt;li&gt;</code>
<code>&lt;main&gt;</code>	<code>&lt;nav&gt;</code>
<code>&lt;noscript&gt;</code>	<code>&lt;ol&gt;</code>
<code>&lt;p&gt;</code>	<code>&lt;pre&gt;</code>
<code>&lt;section&gt;</code>	<code>&lt;table&gt;</code>
<code>&lt;tfoot&gt;</code>	<code>&lt;ul&gt;</code>
<code>&lt;video&gt;</code>	

Here are the inline elements in HTML:

**The `<div>` Element** The `<div>` element is often used as a container for other HTML elements.

The `<div>` element has no required attributes, but **style**, **class** and **id** are

<code>&lt;a&gt;</code>	<code>&lt;abbr&gt;</code>
<code>&lt;acronym&gt;</code>	<code>&lt;b&gt;</code>
<code>&lt;bdo&gt;</code>	<code>&lt;big&gt;</code>
<code>&lt;br&gt;</code>	<code>&lt;button&gt;</code>
<code>&lt;cite&gt;</code>	<code>&lt;code&gt;</code>
<code>&lt;dfn&gt;</code>	<code>&lt;em&gt;</code>
<code>&lt;i&gt;</code>	<code>&lt;img&gt;</code>
<code>&lt;input&gt;</code>	<code>&lt;kbd&gt;</code>
<code>&lt;label&gt;</code>	<code>&lt;map&gt;</code>
<code>&lt;object&gt;</code>	<code>&lt;output&gt;</code>
<code>&lt;q&gt;</code>	<code>&lt;samp&gt;</code>
<code>&lt;script&gt;</code>	<code>&lt;select&gt;</code>
<code>&lt;small&gt;</code>	<code>&lt;span&gt;</code>
<code>&lt;strong&gt;</code>	<code>&lt;sub&gt;</code>
<code>&lt;sup&gt;</code>	<code>&lt;textarea&gt;</code>
<code>&lt;time&gt;</code>	<code>&lt;tt&gt;</code>
<code>&lt;var&gt;</code>	

common.

When used together with CSS, the **<div>** element can be used to style blocks of content:

## Example

```
<div style="background-color:black;color:white;padding:20px;">
 <h2>London</h2>
 <p>London is the capital city of England. It is the most populous city
in the United Kingdom, with a metropolitan area of over 13 million
inhabitants.</p>
</div>
```

## The **<span>** Element

The **<span>** element is an inline container used to mark up a part of a text, or a part of a document.

The **<span>** element has no required attributes, but **style**, **class** and **id** are common.

When used together with CSS, the **<span>** element can be used to style parts of the text:

## Example

```
<p>My mother
has blue ey
es and my father
has dark
green eyes.</p>
```

## Chapter Summary

- There are two display values: block and inline
- A block-level element always starts on a new line and takes up the full width available
- An inline element does not start on a new line and it only takes up as much width as necessary
- The `<div>` element is a block-level and is often used as a container for other HTML elements
- The `<span>` element is an inline container used to mark up a part of a text, or a part of a document

## 19. HTML Class Attribute

The HTML **class** attribute is used to specify a class for an HTML element.

Multiple HTML elements can share the same class.

### Using The class Attribute

The **class** attribute is often used to point to a class name in a style sheet. It can also be used by a JavaScript to access and manipulate elements with the specific class name.

In the following example we have three **<div>** elements with a **class** attribute with the value of "city". All of the three **<div>** elements will be styled equally according to the **.city** style definition in the head section:

### Example

```
<!DOCTYPE html>
<html>
<head>
<style>
.city {
 background-color: tomato;
 color: white;
 border: 2px solid black;
 margin: 20px;
 padding: 20px;
}
</style>
</head>
<body>

<div class="city">
 <h2>London</h2>
 <p>London is the capital of England.</p>
</div>
```

```

<div class="city">
 <h2>Paris</h2>
 <p>Paris is the capital of France.</p>
</div>

<div class="city">
 <h2>Tokyo</h2>
 <p>Tokyo is the capital of Japan.</p>
</div>

</body>
</html>

```

In the following example we have two `<span>` elements with a `class` attribute with the value of "note". Both `<span>` elements will be styled equally according to the `.note` style definition in the head section:

## Example

```

<!DOCTYPE html>
<html>
<head>
<style>
.note {
 font-size: 120%;
 color: red;
}
</style>
</head>
<body>

<h1>My Important Heading</h1>
<p>This is some important text.</p>

</body>
</html>

```

## The Syntax For Class

To create a class; write a period (.) character, followed by a class name. Then, define the CSS properties within curly braces {}:

## Example

Create a class named "city":

```
<!DOCTYPE html>
<html>
<head>
<style>
.city {
 background-color: tomato;
 color: white;
 padding: 10px;
}
</style>
</head>
<body>

<h2 class="city">London</h2>
<p>London is the capital of England.</p>

<h2 class="city">Paris</h2>
<p>Paris is the capital of France.</p>

<h2 class="city">Tokyo</h2>
<p>Tokyo is the capital of Japan.</p>

</body>
</html>
```

## Multiple Classes

HTML elements can belong to more than one class.

To define multiple classes, separate the class names with a space, e.g. `<div class="city main">`. The element will be styled according to all the classes specified.

In the following example, the first `<h2>` element belongs to both the `city` class and also to the `main` class, and will get the CSS styles from both of the classes:

## Example

```
<h2 class="city main">London</h2>
<h2 class="city">Paris</h2>
<h2 class="city">Tokyo</h2>
```

## Different Elements Can Share Same Class

Different HTML elements can point to the same class name.

In the following example, both `<h2>` and `<p>` points to the `"city"` class and will share the same style:

## Example

```
<h2 class="city">Paris</h2>
<p class="city">Paris is the capital of France</p>
```

## Use of The class Attribute in JavaScript

The class name can also be used by JavaScript to perform certain tasks for specific elements.

JavaScript can access elements with a specific class name with the `getElementsByClassName()` method:

## Example

Click on a button to hide all elements with the class name `"city"`:



```
<script>
function myFunction() {
 var x = document.getElementsByClassName("city");
 for (var i = 0; i < x.length; i++) {
 x[i].style.display = "none";
 }
}
</script>
```

## Chapter Summary

- The HTML **class** attribute specifies one or more class names for an element
- Classes are used by CSS and JavaScript to select and access specific elements
- The **class** attribute can be used on any HTML element
- The class name is case sensitive
- Different HTML elements can point to the same class name
- JavaScript can access elements with a specific class name with the **getElementsByClassName()** method

## 20. HTML ID Attribute

The HTML **id** attribute is used to specify a unique id for an HTML element.

You cannot have more than one element with the same id in an HTML document.

### Using The id Attribute

The **id** attribute specifies a unique id for an HTML element. The value of the **id** attribute must be unique within the HTML document.

The **id** attribute is used to point to a specific style declaration in a style sheet. It is also used by JavaScript to access and manipulate the element with the specific id.

The syntax for id is: write a hash character (#), followed by an id name. Then, define the CSS properties within curly braces {}.

In the following example we have an **<h1>** element that points to the id name "myHeader". This **<h1>** element will be styled according to the **#myHeader** style definition in the head section:

### Example

```
<!DOCTYPE html>
<html>
<head>
<style>
#myHeader {
 background-color: lightblue;
 color: black;
 padding: 40px;
 text-align: center;
}
</style>
</head>
```

```
<body>

<h1 id="myHeader">My Header</h1>

</body>
</html>
```

**Note:** The id name is case sensitive!

**Note:** The id name must contain at least one character, and must not contain whitespaces (spaces, tabs, etc.).

## Difference Between Class and ID

A class name can be used by multiple HTML elements, while an id name must only be used by one HTML element within the page:

### Example

```
<style>
/* Style the element with the id "myHeader" */
#myHeader {
 background-color: lightblue;
 color: black;
 padding: 40px;
 text-align: center;
}

/* Style all elements with the class name "city" */
.city {
 background-color: tomato;
 color: white;
 padding: 10px;
}
</style>

<!-- An element with a unique id -->
<h1 id="myHeader">My Cities</h1>

<!-- Multiple elements with same class -->
<h2 class="city">London</h2>
<p>London is the capital of England.</p>

<h2 class="city">Paris</h2>
```

```
<p>Paris is the capital of France.</p>
```

```
<h2 class="city">Tokyo</h2>
```

```
<p>Tokyo is the capital of Japan.</p>
```

## HTML Bookmarks with ID and Links

HTML bookmarks are used to allow readers to jump to specific parts of a webpage.

Bookmarks can be useful if your page is very long.

To use a bookmark, you must first create it, and then add a link to it.

Then, when the link is clicked, the page will scroll to the location with the bookmark.

### Example

First, create a bookmark with the `id` attribute:

```
<h2 id="C4">Chapter 4</h2>
```

Then, add a link to the bookmark ("Jump to Chapter 4"), from within the same page:

```
Jump to Chapter 4
```

Or, add a link to the bookmark ("Jump to Chapter 4"), from another page:

```
Jump to Chapter 4
```

# Using The id Attribute in JavaScript

The **id** attribute can also be used by JavaScript to perform some tasks for that specific element.

JavaScript can access an element with a specific id with the **getElementById()** method:

```
<script>
function displayResult() {
 document.getElementById("myHeader").innerHTML = "Have a nice day!";
}
</script>
```

## Chapter Summary

- The **id** attribute is used to specify a unique id for an HTML element
- The value of the **id** attribute must be unique within the HTML document
- The **id** attribute is used by CSS and JavaScript to style/select a specific element
- The value of the **id** attribute is case sensitive
- The **id** attribute is also used to create HTML bookmarks
- JavaScript can access an element with a specific id with the **getElementById()** method

## 21. HTML Iframes

An HTML iframe is used to display a web page within a web page.

### HTML Iframe Syntax

The HTML `<iframe>` tag specifies an inline frame.

An inline frame is used to embed another document within the current HTML document.

### Syntax

```
<iframe src="url" title="description">
```

**Tip:** It is a good practice to always include a `title` attribute for the `<iframe>`. This is used by screen readers to read out what the content of the iframe is.

### Iframe – Set Height and Width

Use the `height` and `width` attributes to specify the size of the iframe.

The height and width are specified in pixels by default:

### Example

```
<iframe src="demo_iframe.htm" height="200" width="300" title="Iframe Example"></iframe>
```

Or you can add the `style` attribute and use the CSS `height` and `width` properties:

## Example

```
<iframe src="demo_iframe.htm" style="height:200px;width:300px;" title="Iframe Example"></iframe>
```

## Iframe – Remove the Border

By default, an iframe has a border around it.

To remove the border, add the **style** attribute and use the CSS **border** property:

## Example

```
<iframe src="demo_iframe.htm" style="border:none;" title="Iframe Example"></iframe>
```

With CSS, you can also change the size, style and color of the iframe's border:

## Example

```
<iframe src="demo_iframe.htm" style="border:2px solid red;" title="Iframe Example"></iframe>
```

## Iframe – Target for a Link

An iframe can be used as the target frame for a link.

The **target** attribute of the link must refer to the **name** attribute of the iframe:

```
<iframe src="demo_iframe.htm" name="iframe_a" title="Iframe Example"></iframe>
```

```
<p>mrkishorekumar.com</p>
```

## Chapter Summary

- The HTML `<iframe>` tag specifies an inline frame
- The `src` attribute defines the URL of the page to embed
- Always include a `title` attribute (for screen readers)
- The `height` and `width` attributes specifies the size of the iframe
- Use `border:none;` to remove the border around the iframe



## 22. HTML JavaScript

JavaScript makes HTML pages more dynamic and interactive.

### Example

#### My First JavaScript

Click me to display Date and Time

### The HTML `<script>` Tag

The HTML `<script>` tag is used to define a client-side script (JavaScript).

The `<script>` element either contains script statements, or it points to an external script file through the `src` attribute.

Common uses for JavaScript are image manipulation, form validation, and dynamic changes of content.

To select an HTML element, JavaScript most often uses the `document.getElementById()` method.

This JavaScript example writes "Hello JavaScript!" into an HTML element with `id="demo"`:

### Example

```
<script>
document.getElementById("demo").innerHTML = "Hello JavaScript!";
</script>
```

## A Taste of JavaScript

Here are some examples of what JavaScript can do:

### Example

JavaScript can change content:

```
document.getElementById("demo").innerHTML = "Hello JavaScript!";
```

### Example

JavaScript can change styles:

```
document.getElementById("demo").style.fontSize = "25px";
document.getElementById("demo").style.color = "red";
document.getElementById("demo").style.backgroundColor = "yellow";
```

### Example

JavaScript can change attributes:

```
document.getElementById("image").src = "picture.gif";
```

## The HTML <noscript> Tag

The HTML **<noscript>** tag defines an alternate content to be displayed to users that have disabled scripts in their browser or have a browser that doesn't support scripts:

### Example

```
<script>
document.getElementById("demo").innerHTML = "Hello JavaScript!";
</script>
<noscript>Sorry, your browser does not support JavaScript!</noscript>
```

## 23. HTML File Paths

A file path describes the location of a file in a web site's folder structure.

Path	Description
<code>&lt;img src="picture.jpg"&gt;</code>	The "picture.jpg" file is located in the same folder as the current page
<code>&lt;img src="images/picture.jpg"&gt;</code>	The "picture.jpg" file is located in the images folder in the current folder
<code>&lt;img src="/images/picture.jpg"&gt;</code>	The "picture.jpg" file is located in the images folder at the root of the current web
<code>&lt;img src="../../picture.jpg"&gt;</code>	The "picture.jpg" file is located in the folder one level up from the current folder

## HTML File Paths

A file path describes the location of a file in a web site's folder structure. File paths are like an address of file for a web browser. We can link any external resource to add in our HTML file with the help of file paths such as images, file, CSS file, JS file, video, etc.

The src or href attribute requires an attribute to link any external source to HTML file.

File paths are used when linking to external files, like:

- Web pages

- Images
- Style sheets
- JavaScripts

## Absolute File Paths

An absolute file path is the full URL to a file:

### Example

```

```

## Relative File Paths

A relative file path points to a file relative to the current page.

In the following example, the file path points to a file in the images folder located at the root of the current web:

```

```

In the following example, the file path points to a file in the images folder located in the current folder:

### Example

```

```

In the following example, the file path points to a file in the images folder located in the folder one level up from the current folder:

### Example

```

```

## Best Practice

It is best practice to use relative file paths (if possible).

When using relative file paths, your web pages will not be bound to your current base URL. All links will work on your own computer (localhost) as well as on your current public domain and your future public domains.

## Important Points for File path:

- Always remember to use proper URL, file name, image name, else it will not display on the webpage.
- Try to use relative file paths, so that your code will be independent of URL.

## 24. HTML – The Head Element

The HTML `<head>` element is a container for the following elements: `<title>`, `<style>`, `<meta>`, `<link>`, `<script>`, and `<base>`.

### The HTML `<head>` Element

The `<head>` element is a container for metadata (data about data) and is placed between the `<html>` tag and the `<body>` tag.

HTML metadata is data about the HTML document. Metadata is not displayed.

Metadata typically define the document title, character set, styles, scripts, and other meta information.

### The HTML `<title>` Element

The `<title>` element defines the title of the document. The title must be text-only, and it is shown in the browser's title bar or in the page's tab.

The `<title>` element is required in HTML documents!

The contents of a page title is very important for search engine optimization (SEO)! The page title is used by search engine algorithms to decide the order when listing pages in search results.

The `<title>` element:

- defines a title in the browser toolbar

- provides a title for the page when it is added to favorites
- displays a title for the page in search engine-results

So, try to make the title as accurate and meaningful as possible!

A simple HTML document:

## Example

```
<!DOCTYPE html>
<html>
<head>
 <title>A Meaningful Page Title</title>
</head>
<body>
```

The content of the document.....

```
</body>
</html>
```

## The HTML <style> Element

The **<style>** element is used to define style information for a single HTML page:

## Example

```
<style>
 body {background-color: powderblue;}
 h1 {color: red;}
 p {color: blue;}
</style>
```

## The HTML <link> Element

The **<link>** element defines the relationship between the current document and an external resource.

The **<link>** tag is most often used to link to external style sheets:

## Example

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

## The HTML **<meta>** Element

The **<meta>** element is typically used to specify the character set, page description, keywords, author of the document, and viewport settings.

The metadata will not be displayed on the page, but are used by browsers (how to display content or reload page), by search engines (keywords), and other web services.

## Examples

**Define the character set used:**

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

**Define keywords for search engines:**

```
<meta name="keywords" content="HTML, CSS, JavaScript">
```

**Define a description of your web page:**

```
<meta name="description" content="Free Web tutorials">
```

**Define the author of a page:**

```
<meta name="author" content="John Doe">
```

**Refresh document every 30 seconds:**

```
<meta http-equiv="refresh" content="30">
```

**Setting the viewport to make your website look good on all devices:**



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

Example of **<meta>** tags:

## Example

```
<meta charset="UTF-8">
<meta name="description" content="Free Web tutorials">
<meta name="keywords" content="HTML, CSS, JavaScript">
<meta name="author" content="John Doe">
```

## Setting The Viewport

The viewport is the user's visible area of a web page. It varies with the device - it will be smaller on a mobile phone than on a computer screen.

You should include the following **<meta>** element in all your web pages:

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

This gives the browser instructions on how to control the page's dimensions and scaling.

The **width=device-width** part sets the width of the page to follow the screen-width of the device (which will vary depending on the device).

The **initial-scale=1.0** part sets the initial zoom level when the page is first loaded by the browser.

Here is an example of a web page without the viewport meta tag, and the same web page with the viewport meta tag:

**Tip:** If you are browsing this page with a phone or a tablet, you can click on the two links below to see the difference.

## The HTML `<script>` Element

The `<script>` element is used to define client-side JavaScripts.

The following JavaScript writes "Hello JavaScript!" into an HTML element with id="demo":

### Example

```
<script>
function myFunction() {
 document.getElementById("demo").innerHTML = "Hello JavaScript!";
}
</script>
```

## The HTML `<base>` Element

The `<base>` element specifies the base URL and/or target for all relative URLs in a page.

The `<base>` tag must have either an href or a target attribute present, or both.

There can only be one single `<base>` element in a document!

### Example

Specify a default URL and a default target for all links on a page:

```
<head>
<base href="https://www.mrkishorekumar.com/" target="_blank">
</head>
```

```
<body>

HTML base Tag
</body>
```

## Chapter Summary

- The **<head>** element is a container for metadata (data about data)
- The **<head>** element is placed between the **<html>** tag and the **<body>** tag
- The **<title>** element is required and it defines the title of the document
- The **<style>** element is used to define style information for a single document
- The **<link>** tag is most often used to link to external style sheets
- The **<meta>** element is typically used to specify the character set, page description, keywords, author of the document, and viewport settings
- The **<script>** element is used to define client-side JavaScripts
- The **<base>** element specifies the base URL and/or target for all relative URLs in a page

## HTML head Elements

Tag	Description
-----	-------------

<code>&lt;head&gt;</code>	Defines information about the document
<code>&lt;title&gt;</code>	Defines the title of a document
<code>&lt;base&gt;</code>	Defines a default address or a default target for all links on a page
<code>&lt;link&gt;</code>	Defines the relationship between a document and an external resource
<code>&lt;meta&gt;</code>	Defines metadata about an HTML document
<code>&lt;script&gt;</code>	Defines a client-side script
<code>&lt;style&gt;</code>	Defines style information for a document

## 25. HTML Layout Elements and Techniques

Websites often display content in multiple columns (like a magazine or a newspaper).

### Example

## Cities

- London
- Paris
- Tokyo

## London

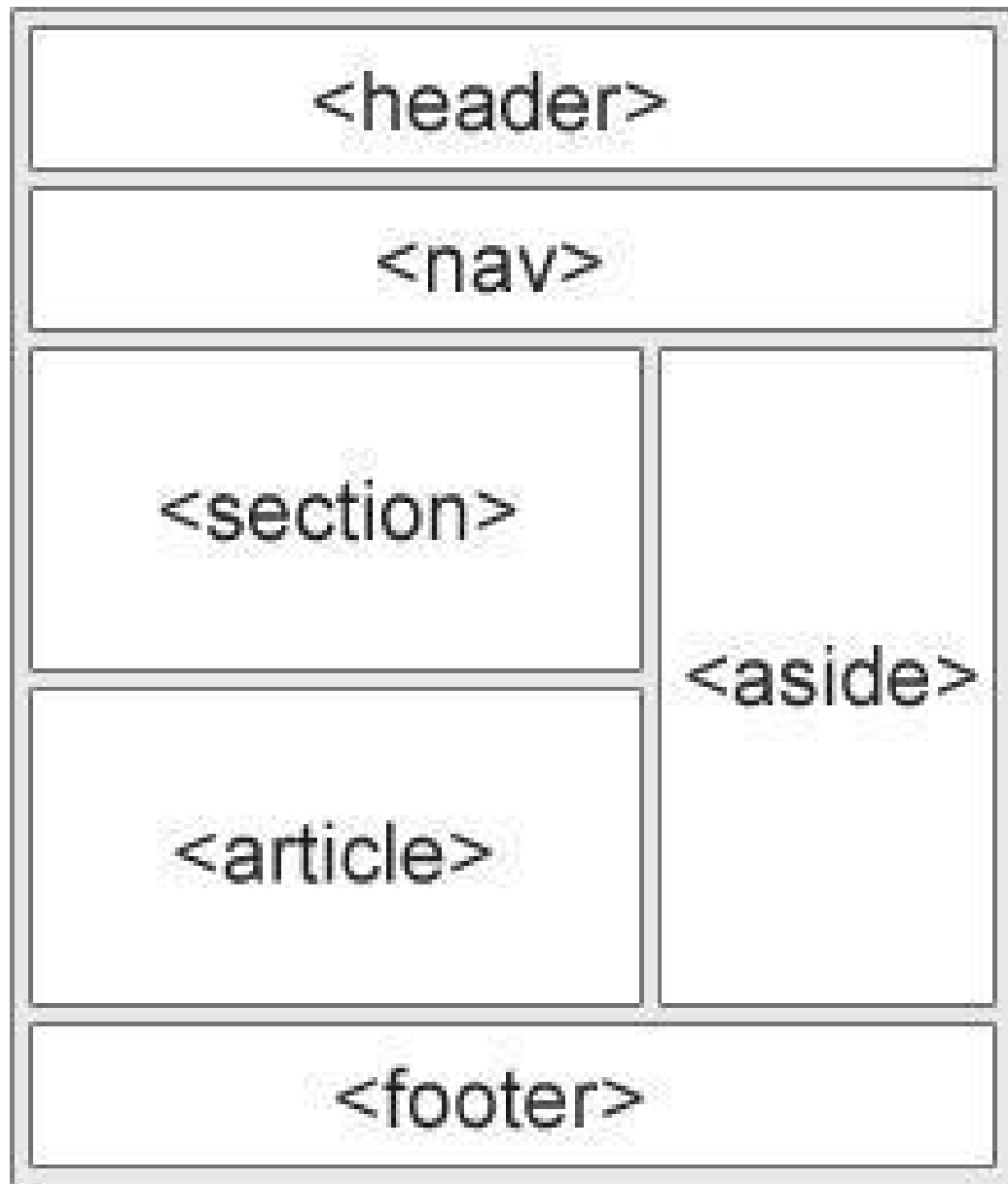
London is the capital city of England. It is the most populous city in the United Kingdom, with a metropolitan area of over 13 million inhabitants.

Standing on the River Thames, London has been a major settlement for two millennia, its history going back to its founding by the Romans, who named it Londinium.

Footer

## HTML Layout Elements

HTML has several semantic elements that define the different parts of a web page:



`<header>` - Defines a header for a document or a section

`<nav>` - Defines a set of navigation links

`<section>` - Defines a section in a document

`<article>` - Defines an independent, self-contained content

`<aside>` - Defines content aside from the content (like a sidebar)

`<footer>` - Defines a footer for a document or a section

**<details>** - Defines additional details that the user can open and close on demand

**<summary>** - Defines a heading for the **<details>** element

## HTML Layout Techniques

There are four different techniques to create multicolumn layouts. Each technique has its pros and cons:

- CSS framework
- CSS float property
- CSS flexbox
- CSS grid

## CSS Float Layout

It is common to do entire web layouts using the CSS **float** property. Float is easy to learn - you just need to remember how the **float** and **clear** properties work. Disadvantages: Floating elements are tied to the document flow, which may harm the flexibility.

## CSS Flexbox Layout

Use of flexbox ensures that elements behave predictably when the page layout must accommodate different screen sizes and different display devices.

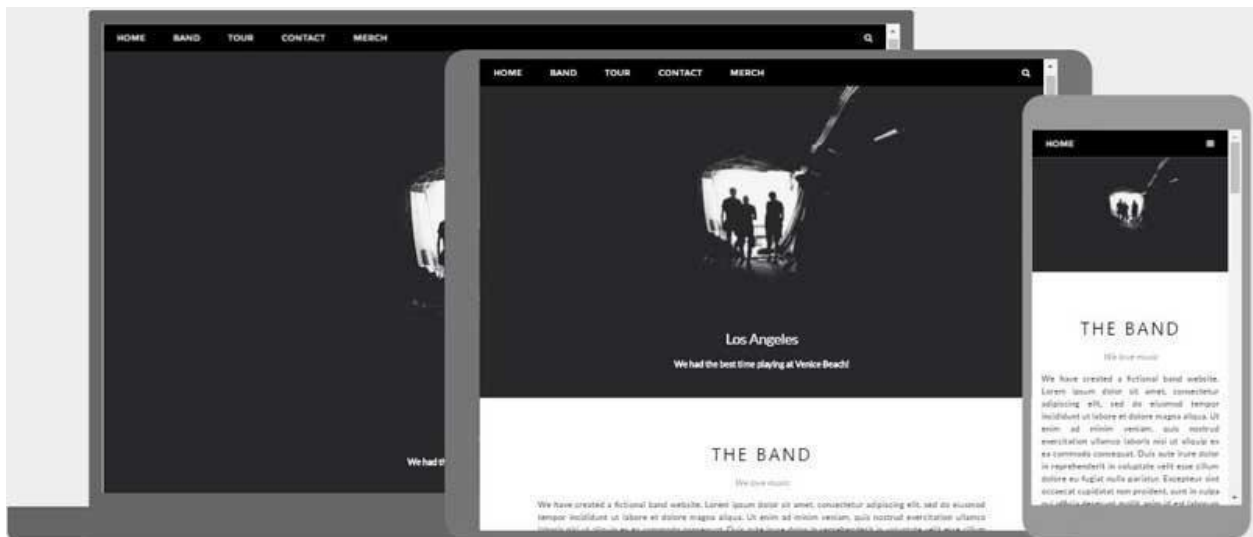
## CSS Grid Layout

The CSS Grid Layout Module offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning.

# 26. HTML Responsive Web Design

Responsive web design is about creating web pages that look good on all devices!

A responsive web design will automatically adjust for different screen sizes and viewports.



## What is Responsive Web Design?

Responsive Web Design is about using HTML and CSS to automatically resize, hide, shrink, or enlarge, a website, to make it look good on all devices (desktops, tablets, and phones):

## Setting The Viewport

To create a responsive website, add the following **<meta>** tag to all your web pages:



## Example

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

This will set the viewport of your page, which will give the browser instructions on how to control the page's dimensions and scaling.

Here is an example of a web page without the viewport meta tag, and the same web page with the viewport meta tag:

Visit : <https://mrkishorekumar.github.io/info/>

## Responsive Images

Responsive images are images that scale nicely to fit any browser size.

### Using the width Property

If the CSS **width** property is set to 100%, the image will be responsive and scale up and down.

## Example

```

```

Notice that in the example above, the image can be scaled up to be larger than its original size. A better solution, in many cases, will be to use the **max-width** property instead.

## Using the max-width Property

If the **max-width** property is set to 100%, the image will scale down if it has to, but never scale up to be larger than its original size:

### Example

```
<picture>
 <source srcset="img_smallflower.jpg" media="(max-width: 600px)">
 <source srcset="img_flowers.jpg" media="(max-width: 1500px)">
 <source srcset="flowers.jpg">

</picture>
```

## Responsive Text Size

The text size can be set with a "vw" unit, which means the "viewport width".

That way the text size will follow the size of the browser window:

### Hello World

Resize the browser window to see how the text size scales.

### Example

```
<h1 style="font-size:10vw">Hello World</h1>
```

Viewport is the browser window size. 1vw = 1% of viewport width. If the viewport is 50cm wide, 1vw is 0.5cm.

## Media Queries

In addition to resize text and images, it is also common to use media queries in responsive web pages.

With media queries you can define completely different styles for different browser sizes.

Example: resize the browser window to see that the three div elements below will display horizontally on large screens and stacked vertically on small screens:

Left Menu

Main Content

Right Content

## Example

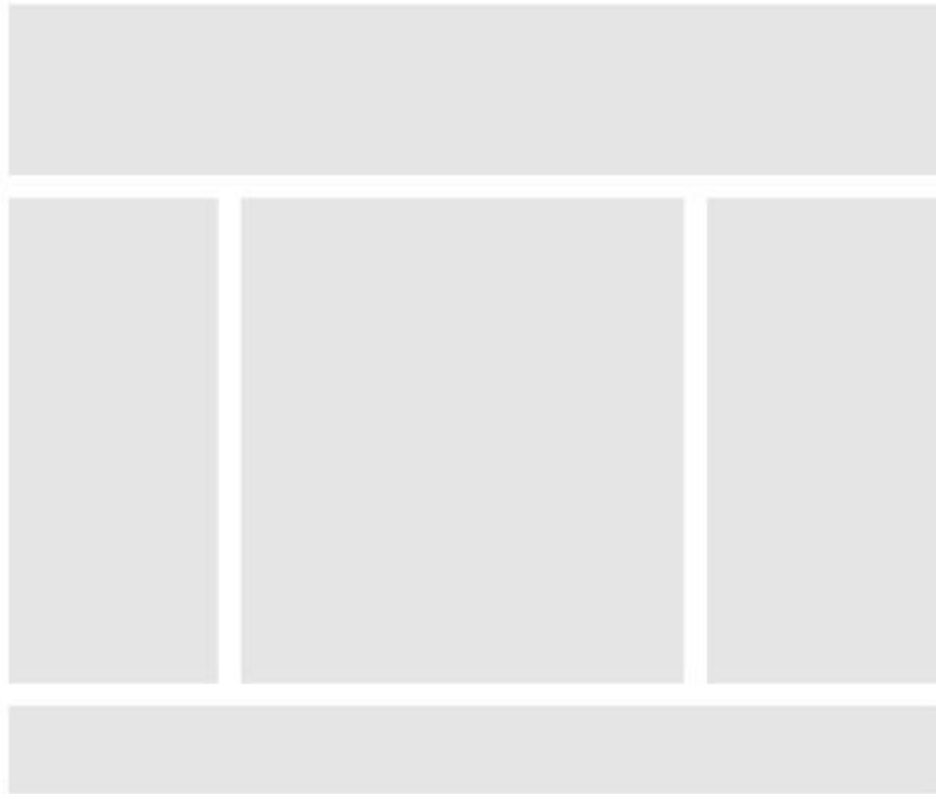
```
<style>
.left, .right {
 float: left;
 width: 20%; /* The width is 20%, by default */
}

.main {
 float: left;
 width: 60%; /* The width is 60%, by default */
}

/* Use a media query to add a breakpoint at 800px: */
@media screen and (max-width: 800px) {
 .left, .main, .right {
 width: 100%; /* The width is 100%, when the viewport is 800px or
smaller */
 }
}
</style>
```

## Responsive Web Page – Full Example

A responsive web page should look good on large desktop screens and on small mobile phones.



## Bootstrap

Another popular CSS framework is Bootstrap. Bootstrap uses HTML, CSS and jQuery to make responsive web pages.

## Example

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```

<title>Bootstrap Example</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</head>
<body>

<div class="container">
 <div class="jumbotron">
 <h1>My First Bootstrap Page</h1>
 </div>
 <div class="row">
 <div class="col-sm-4">
 ...
 </div>
 <div class="col-sm-4">
 ...
 </div>
 <div class="col-sm-4">
 ...
 </div>
 </div>
</div>

</body>
</html>

```

## 27. HTML Computer Code Elements

HTML contains several elements for defining user input and computer code.

### Example

```
<code>
x = 5;
y = 6;
z = x + y;
</code>
```

### HTML <kbd> For Keyboard Input

The HTML **<kbd>** element is used to define keyboard input. The content inside is displayed in the browser's default monospace font.

### Example

Define some text as keyboard input in a document:

```
<p>Save the document by pressing <kbd>Ctrl + S</kbd></p>
```

Result:

Save the document by pressing Ctrl + S

### HTML <samp> For Program Output

The HTML **<samp>** element is used to define sample output from a computer program. The content inside is displayed in the browser's default monospace font.

## Example

Define some text as sample output from a computer program in a document:

```
<p>Message from my computer:</p>
<p><samp>File not found.
Press F1 to continue</samp></p>
```

### Result:

Message from my computer:

```
File not found.
Press F1 to continue
```

## HTML `<code>` For Computer Code

The HTML `<code>` element is used to define a piece of computer code. The content inside is displayed in the browser's default monospace font.

## Example

Define some text as computer code in a document:

```
<code>
x = 5;
y = 6;
z = x + y;
</code>
```

### Result:

```
x = 5; y = 6; z = x + y;
```

Notice that the `<code>` element does not preserve extra whitespace and line-breaks.

To fix this, you can put the `<code>` element inside a `<pre>` element:

## Example

```
<pre>
<code>
x = 5;
y = 6;
z = x + y;
</code>
</pre>
```

### Result:

```
x = 5;
y = 6;
z = x + y;
```

## HTML `<var>` For Variables

The HTML `<var>` element is used to define a variable in programming or in a mathematical expression. The content inside is typically displayed in italic.

## Example

Define some text as variables in a document:

```
<p>The area of a triangle is: 1/2
x <var>b</var> x <var>h</var>, where <var>b</var> is the
base, and <var>h</var> is the vertical height.</p>
```

### Result:

The area of a triangle is:  $1/2 \times b \times h$ , where  $b$  is the base, and  $h$  is the vertical height.



## Chapter Summary

- The `<kbd>` element defines keyboard input
- The `<samp>` element defines sample output from a computer program
- The `<code>` element defines a piece of computer code
- The `<var>` element defines a variable in programming or in a mathematical expression
- The `<pre>` element defines preformatted text

## 28. HTML Semantic Elements

Semantic elements = elements with a meaning.

### What are Semantic Elements?

A semantic element clearly describes its meaning to both the browser and the developer.

Examples of **non-semantic** elements: `<div>` and `<span>` - Tells nothing about its content.

Examples of **semantic** elements: `<form>`, `<table>`, and `<article>` - Clearly defines its content.

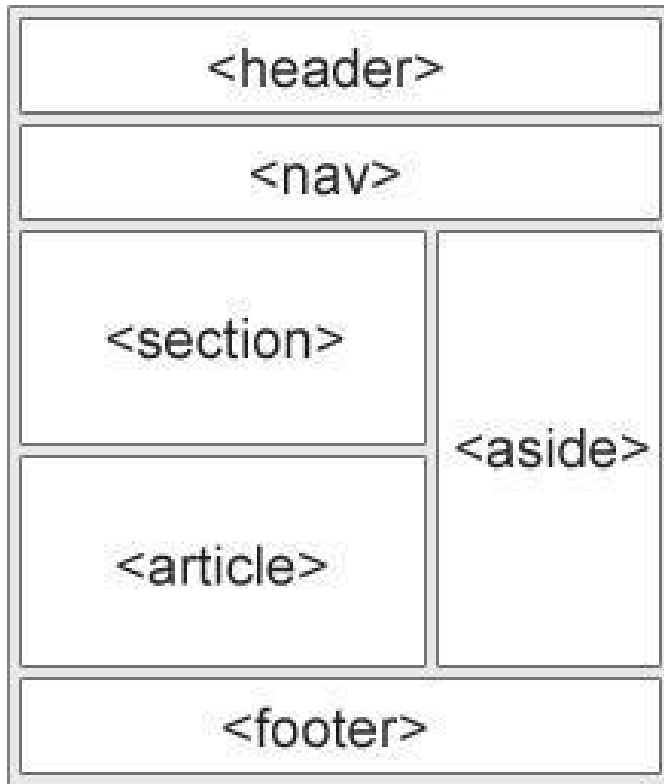
### Semantic Elements in HTML

Many web sites contain HTML code like: `<div id="nav">` `<div class="header">` `<div id="footer">` to indicate navigation, header, and footer.

In HTML there are some semantic elements that can be used to define different parts of a web page:

- `<article>`
- `<aside>`
- `<details>`
- `<figcaption>`
- `<figure>`
- `<footer>`
- `<header>`
- `<main>`
- `<mark>`

- `<nav>`
- `<section>`
- `<summary>`
- `<time>`



## HTML `<section>` Element

The `<section>` element defines a section in a document.

According to W3C's HTML documentation: "A section is a thematic grouping of content, typically with a heading."

A web page could normally be split into sections for introduction, content, and contact information.

## Example

Two sections in a document:

```
<section>
<h1>WWF</h1>
<p>The World Wide Fund for Nature (WWF) is an international
organization working on issues regarding the conservation, research
and restoration of the environment, formerly named the World Wildlife
Fund. WWF was founded in 1961.</p>
</section>
```

```
<section>
<h1>WWF's Panda symbol</h1>
<p>The Panda has become the symbol of WWF. The well-known panda logo
of WWF originated from a panda named Chi Chi that was transferred from
the Beijing Zoo to the London Zoo in the same year of the
establishment of WWF.</p>
</section>
```

## HTML <article> Element

The **<article>** element specifies independent, self-contained content.

An article should make sense on its own, and it should be possible to distribute it independently from the rest of the web site.

Examples of where an **<article>** element can be used:

- Forum post
- Blog post
- Newspaper article

## Example

Three articles with independent, self-contained content:

```
<article>
<h2>Google Chrome</h2>
<p>Google Chrome is a web browser developed by Google, released in
2008. Chrome is the world's most popular web browser today!</p>
</article>

<article>
<h2>Mozilla Firefox</h2>
<p>Mozilla Firefox is an open-source web browser developed by Mozilla.
Firefox has been the second most popular web browser since January,
2018.</p>
</article>

<article>
<h2>Microsoft Edge</h2>
<p>Microsoft Edge is a web browser developed by Microsoft, released in
2015. Microsoft Edge replaced Internet Explorer.</p>
</article>
```

## Example 2

Use CSS to style the <article> element:

```
<html>
<head>
<style>
.all-browsers {
 margin: 0;
 padding: 5px;
 background-color: lightgray;
}

.all-browsers > h1, .browser {
 margin: 10px;
 padding: 5px;
}
```

```

.browser {
 background: white;
}

.browser > h2, p {
 margin: 4px;
 font-size: 90%;
}
</style>
</head>
<body>

<article class="all-browsers">
 <h1>Most Popular Browsers</h1>
 <article class="browser">
 <h2>Google Chrome</h2>
 <p>Google Chrome is a web browser developed by Google,
released in 2008. Chrome is the world's most popular web
browser today!</p>
 </article>
 <article class="browser">
 <h2>Mozilla Firefox</h2>
 <p>Mozilla Firefox is an open-source web browser
developed by Mozilla. Firefox has been the second most
popular web browser since January, 2018.</p>
 </article>
 <article class="browser">
 <h2>Microsoft Edge</h2>
 <p>Microsoft Edge is a web browser developed by
Microsoft, released in 2015. Microsoft Edge replaced
Internet Explorer.</p>
 </article>
</article>

</body>
</html>

```

## Nesting <article> in <section> or Vice Versa?

The <article> element specifies independent, self-contained content.

The <section> element defines section in a document.

Can we use the definitions to decide how to nest those elements? No, we cannot!

So, you will find HTML pages with <section> elements containing <article> elements, and <article> elements containing <section> elements.

## HTML <header> Element

The <header> element represents a container for introductory content or a set of navigational links.

A <header> element typically contains:

- one or more heading elements (<h1> - <h6>)
- logo or icon
- authorship information

**Note:** You can have several <header> elements in one HTML document. However, <header> cannot be placed within a <footer>, <address> or another <header> element.

## Example

A header for an <article>:

```
<article>
 <header>
 <h1>What Does WWF Do?</h1>
```

```
<p>WWF's mission:</p>
</header>
<p>WWF's mission is to stop the degradation of our planet's natural
environment,
and build a future in which humans live in harmony with nature.</p>
</article>
```

## HTML <footer> Element

The <footer> element defines a footer for a document or section.

A <footer> element typically contains:

- authorship information
- copyright information
- contact information
- sitemap
- back to top links
- related documents

You can have several <footer> elements in one document.

### Example

A footer section in a document:

```
<footer>
 <p>Author: Hege Refsnes</p>
 <p>hege@example.com</p>
</footer>
```

## HTML <nav> Element

The <nav> element defines a set of navigation links.



Notice that NOT all links of a document should be inside a `<nav>` element. The `<nav>` element is intended only for major block of navigation links.

Browsers, such as screen readers for disabled users, can use this element to determine whether to omit the initial rendering of this content.

## Example

A set of navigation links:

```
<nav>
 HTML |
 CSS |
 JavaScript |
 jQuery
</nav>
```

## HTML `<aside>` Element

The `<aside>` element defines some content aside from the content it is placed in (like a sidebar).

The `<aside>` content should be indirectly related to the surrounding content.

## Example

Display some content aside from the content it is placed in:

```
<p>My family and I visited The Epcot center this summer. The weather
was nice, and Epcot was amazing! I had a great summer together with my
family!</p>
```

```
<aside>
<h4>Epcot Center</h4>
<p>Epcot is a theme park at Walt Disney World Resort featuring
exciting attractions, international pavilions, award-winning fireworks
```

```
and seasonal special events.</p>
</aside>
```

## Example 2

Use CSS to style the <aside> element:

```
<html>
<head>
<style>
aside {
 width: 30%;
 padding-left: 15px;
 margin-left: 15px;
 float: right;
 font-style: italic;
 background-color: lightgray;
}
</style>
</head>
<body>

<p>My family and I visited The Epcot center this summer. The weather was
nice, and Epcot was amazing! I had a great summer together with my
family!</p>

<aside>
<p>The Epcot center is a theme park at Walt Disney World Resort featuring
exciting attractions, international pavilions, award-winning fireworks and
seasonal special events.</p>
</aside>

<p>My family and I visited The Epcot center this summer. The weather was
nice, and Epcot was amazing! I had a great summer together with my
family!</p>
<p>My family and I visited The Epcot center this summer. The weather was
nice, and Epcot was amazing! I had a great summer together with my
family!</p>

</body>
</html>
```

# HTML <figure> and <figcaption> Elements

The <figure> tag specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.

The <figcaption> tag defines a caption for a <figure> element.

The <figcaption> element can be placed as the first or as the last child of a <figure> element.

The <img> element defines the actual image/illustration.

## Example

```
<figure>

 <figcaption>Fig1. - Trulli, Puglia, Italy.</figcaption>
</figure>
```

# Semantic Elements in HTML

Below is a list of some of the semantic elements in HTML.

Value	Description
<article>	Defines independent, self-contained content
<aside>	Defines content aside from the page content
<details>	Defines additional details that the user can view or hide
<figcaption>	Defines a caption for a <figure> element
<figure>	Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.
<footer>	Defines a footer for a document or section
<header>	Specifies a header for a document or section
<main>	Specifies the main content of a document
<mark>	Defines marked/highlighted text
<nav>	Defines navigation links
<section>	Defines a section in a document
<summary>	Defines a visible heading for a <details> element
<time>	Defines a date/time



## 29. HTML Style Guide and Coding Conventions

A consistent, clean, and tidy HTML code makes it easier for others to read and understand your code.

Here are some guidelines and tips for creating good HTML code.

### Always Declare Document Type

Always declare the document type as the first line in your document.

The correct document type for HTML is:

```
<!DOCTYPE html>
```

### Use Lowercase Element Names

HTML allows mixing uppercase and lowercase letters in element names.

However, we recommend using lowercase element names, because:

- Mixing uppercase and lowercase names looks bad
- Developers normally use lowercase names
- Lowercase looks cleaner
- Lowercase is easier to write

#### Good:

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

#### Bad:

```
<BODY>
<P>This is a paragraph.</P>
</BODY>
```

## Close All HTML Elements

In HTML, you do not have to close all elements (for example the `<p>` element).

However, we strongly recommend closing all HTML elements, like this:

### Good:

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

### Bad:

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

## Use Lowercase Attribute Names

HTML allows mixing uppercase and lowercase letters in attribute names.

However, we recommend using lowercase attribute names, because:

- Mixing uppercase and lowercase names looks bad
- Developers normally use lowercase names
- Lowercase look cleaner
- Lowercase are easier to write

### Good:

```
Visit our HTML
tutorial
```

## Bad:

```
Visit our HTML
tutorial
```

## Always Quote Attribute Values

HTML allows attribute values without quotes.

However, we recommend quoting attribute values, because:

- Developers normally quote attribute values
- Quoted values are easier to read
- You MUST use quotes if the value contains spaces

## Good:

```
<table class="striped">
```

## Bad:

```
<table class=striped>
```

## Very bad:

This will not work, because the value contains spaces:

```
<table class=table striped>
```

## Always Specify alt, width, and height for Images

Always specify the **alt** attribute for images. This attribute is important if the image for some reason cannot be displayed.

Also, always define the **width** and **height** of images. This reduces flickering, because the browser can reserve space for the image before loading.

## Good:

```

```



**Bad:**

```

```

## Spaces and Equal Signs

HTML allows spaces around equal signs. But space-less is easier to read and groups entities better together.

**Good:**

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

**Bad:**

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

## Avoid Long Code Lines

When using an HTML editor, it is NOT convenient to scroll right and left to read the HTML code.

Try to avoid too long code lines.

## Blank Lines and Indentation

Do not add blank lines, spaces, or indentations without a reason.

For readability, add blank lines to separate large or logical code blocks.

For readability, add two spaces of indentation. Do not use the tab key.

**Good:**

```
<body>
```

```
<h1>Famous Cities</h1>
```

```
<h2>Tokyo</h2>
```

```
<p>Tokyo is the capital of Japan, the center of the Greater Tokyo
Area,
```

and the most populous metropolitan area in the world.  
It is the seat of the Japanese government and the Imperial Palace,  
and the home of the Japanese Imperial Family.</p>

</body>

## Bad:

<body>

<h1>Famous Cities</h1>

<h2>Tokyo</h2>

<p>

Tokyo is the capital of Japan, the center of the Greater Tokyo  
Area,

and the most populous metropolitan area in the world.

It is the seat of the Japanese government and the Imperial Palace,  
and the home of the Japanese Imperial Family.

</p>

</body>

## Good Table Example:

<table>

<tr>

<th>Name</th>

<th>Description</th>

</tr>

<tr>

<td>A</td>

<td>Description of A</td>

</tr>

<tr>

<td>B</td>

<td>Description of B</td>

</tr>

</table>

## Good List Example:

```

 London
 Paris
 Tokyo

```

## Never Skip the <title> Element

The `<title>` element is required in HTML.

The contents of a page title is very important for search engine optimization (SEO)! The page title is used by search engine algorithms to decide the order when listing pages in search results.

The `<title>` element:

- defines a title in the browser toolbar
- provides a title for the page when it is added to favorites
- displays a title for the page in search-engine results

So, try to make the title as accurate and meaningful as possible:

```
<title>HTML Style Guide and Coding Conventions</title>
```

## Omitting <html> and <body>?

An HTML page will validate without the `<html>` and `<body>` tags:

## Example

```
<!DOCTYPE html>
<head>
 <title>Page Title</title>
</head>

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

However, we strongly recommend to always add the `<html>` and `<body>` tags!

Omitting `<body>` can produce errors in older browsers.

Omitting `<html>` and `<body>` can also crash DOM and XML software.

## Omitting `<head>`?

The HTML `<head>` tag can also be omitted.

Browsers will add all elements before `<body>`, to a default `<head>` element.

## Example

```
<!DOCTYPE html>
<html>
<title>Page Title</title>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

However, we recommend using the `<head>` tag.

## Close Empty HTML Elements?

In HTML, it is optional to close empty elements.

### Allowed:

```
<meta charset="utf-8">
```

## Also Allowed:

```
<meta charset="utf-8" />
```

If you expect XML/XHTML software to access your page, keep the closing slash (/), because it is required in XML and XHTML.

## Add the lang Attribute

You should always include the **lang** attribute inside the **<html>** tag, to declare the language of the Web page. This is meant to assist search engines and browsers.

## Example

```
<!DOCTYPE html>
<html lang="en-us">
<head>
 <title>Page Title</title>
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

## Meta Data

To ensure proper interpretation and correct search engine indexing, both the language and the character encoding **<meta charset="charset">** should be defined as early as possible in an HTML document:

```
<!DOCTYPE html>
<html lang="en-us">
<head>
 <meta charset="UTF-8">
```

```
<title>Page Title</title>
</head>
```

## Setting The Viewport

The viewport is the user's visible area of a web page. It varies with the device - it will be smaller on a mobile phone than on a computer screen.

You should include the following `<meta>` element in all your web pages:

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

This gives the browser instructions on how to control the page's dimensions and scaling.

The `width=device-width` part sets the width of the page to follow the screen-width of the device (which will vary depending on the device).

The `initial-scale=1.0` part sets the initial zoom level when the page is first loaded by the browser.

Here is an example of a web page *without* the viewport meta tag, and the same web page *with* the viewport meta tag:

**Tip:** If you are browsing this page with a phone or a tablet, you can click on the two links below to see the difference.

## HTML Comments

Short comments should be written on one line, like this:

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

Comments that spans more than one line, should be written like this:

```
<!--
 This is a long comment example. This is a long comment example.
 This is a long comment example. This is a long comment example.
-->
```

Long comments are easier to observe if they are indented with two spaces.

## Using Style Sheets

Use simple syntax for linking to style sheets (the **type** attribute is not necessary):

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

Short CSS rules can be written compressed, like this:

```
p.intro {font-family:Verdana;font-size:16em;}
```

Long CSS rules should be written over multiple lines:

```
body {
 background-color: lightgrey;
 font-family: "Arial Black", Helvetica, sans-serif;
 font-size: 16em;
 color: black;
}
```

- Place the opening bracket on the same line as the selector
- Use one space before the opening bracket
- Use two spaces of indentation
- Use semicolon after each property-value pair, including the last
- Only use quotes around values if the value contains spaces
- Place the closing bracket on a new line, without leading spaces

## Loading JavaScript in HTML

Use simple syntax for loading external scripts (the **type** attribute is not necessary):

Use simple syntax for loading external scripts (the **type** attribute is not necessary):

```
<script src="myscript.js">
```

## Accessing HTML Elements with JavaScript

Using "untidy" HTML code can result in JavaScript errors.

These two JavaScript statements will produce different results:

### Example

```
getElementById("Demo").innerHTML = "Hello";
```

```
getElementById("demo").innerHTML = "Hello";
```

## Use Lower Case File Names

Some web servers (Apache, Unix) are case sensitive about file names: "london.jpg" cannot be accessed as "London.jpg".

Other web servers (Microsoft, IIS) are not case sensitive: "london.jpg" can be accessed as "London.jpg".

If you use a mix of uppercase and lowercase, you have to be aware of this.

If you move from a case-insensitive to a case-sensitive server, even small errors will break your web!

To avoid these problems, always use lowercase file names!

## File Extensions

HTML files should have a **.html** extension (**.htm** is allowed).



CSS files should have a **.css** extension.

JavaScript files should have a **.js** extension.

## Differences Between .htm and .html?

There is no difference between the .htm and .html file extensions!

Both will be treated as HTML by any web browser and web server.

## Default Filenames

When a URL does not specify a filename at the end (like "https://www.sample.com/"), the server just adds a default filename, such as "index.html", "index.htm", "default.html", or "default.htm".

If your server is configured only with "index.html" as the default filename, your file must be named "index.html", and not "default.html".

However, servers can be configured with more than one default filename; usually you can set up as many default filenames as you want.

## 30. HTML Entities

Reserved characters in HTML must be replaced with character entities.

### HTML Entities

Some characters are reserved in HTML.

If you use the less than (<) or greater than (>) signs in your text, the browser might mix them with tags.

Character entities are used to display reserved characters in HTML.

A character entity looks like this:

*&entity\_name;*

OR

*&#entity\_number;*

To display a less than sign (<) we must write: **&lt;** or **&#60;**;

**Advantage of using an entity name: An entity name is easy to remember.**

**Disadvantage of using an entity name: Browsers may not support all entity names, but the support for entity numbers is good.**

### Non-breaking Space

A commonly used entity in HTML is the non-breaking space: **&nbsp;**;

A non-breaking space is a space that will not break into a new line.

Two words separated by a non-breaking space will stick together (not break into a new line). This is handy when breaking the words might be disruptive.

Examples:

- § 10
- 10 km/h
- 10 PM


Another common use of the non-breaking space is to prevent browsers from truncating spaces in HTML pages.

If you write 10 spaces in your text, the browser will remove 9 of them. To add real spaces to your text, you can use the **&nbsp;** character entity.

**Tip:** The non-breaking hyphen (&#8209;) is used to define a hyphen character (-) that does not break into a new line.

## Some Useful HTML Character Entities

Result	Description	Entity Name	Entity Number
	non-breaking space	&nbsp;	&#160;
<	less than	&lt;	&#60;
>	greater than	&gt;	&#62;
&	ampersand	&amp;	&#38;
¢	cent	&cent;	&#162;
£	pound	&pound;	&#163;
¥	yen	&yen;	&#165;
€	euro	&euro;	&#8364;
©	copyright	&copy;	&#169;
®	registered trademark	&reg;	&#174;

 Entity names are case sensitive.

## Combining Diacritical Marks

A diacritical mark is a "glyph" added to a letter.

Some diacritical marks, like grave ( ` ) and acute ( ´ ) are called accents.

Diacritical marks can appear both above and below a letter, inside a letter, and between two letters.

Diacritical marks can be used in combination with alphanumeric characters to produce a character that is not present in the character set (encoding) used in the page.

Here are some examples:

Mark	Character	Construct	Result
·	a	a&#768;	à
ˆ	a	a&#769;	â
ˆ	a	a&#770;	ã
˜	a	a&#771;	ä
·	O	O&#768;	Ò
ˆ	O	O&#769;	Ó
ˆ	O	O&#770;	Ô
˜	O	O&#771;	Õ

# 31. HTML Symbols

Symbols that are not present on your keyboard can also be added by using entities.

## HTML Symbol Entities

HTML entities were described in the previous chapter.

Many mathematical, technical, and currency symbols are not present on a normal keyboard.

To add such symbols to an HTML page, you can use the entity name or the entity number (a decimal or a hexadecimal reference) for the symbol.

### Example

Display the euro sign, €, with an entity name, a decimal, and a hexadecimal value:

```
<p>I will display €</p>
<p>I will display €</p>
<p>I will display €</p>
```

**Will display as:**

I will display €

I will display €

I will display €

## Some Mathematical Symbols Supported by HTML

Char	Number	Entity	Description
$\forall$	&#8704;	&forall;	FOR ALL
$\partial$	&#8706;	&part;	PARTIAL DIFFERENTIAL
$\exists$	&#8707;	&exist;	THERE EXISTS
$\emptyset$	&#8709;	&empty;	EMPTY SETS
$\nabla$	&#8711;	&nabla;	NABLA
$\in$	&#8712;	&isin;	ELEMENT OF
$\notin$	&#8713;	&notin;	NOT AN ELEMENT OF
$\ni$	&#8715;	&ni;	CONTAINS AS MEMBER
$\prod$	&#8719;	&prod;	N-ARY PRODUCT
$\sum$	&#8721;	&sum;	N-ARY SUMMATION

## Some Greek Letters & Other Entities Supported by HTML

Char	Number	Entity	Description
A	&#913;	&Alpha;	GREEK CAPITAL LETTER ALPHA
B	&#914;	&Beta;	GREEK CAPITAL LETTER BETA
Γ	&#915;	&Gamma;	GREEK CAPITAL LETTER GAMMA
Δ	&#916;	&Delta;	GREEK CAPITAL LETTER DELTA
E	&#917;	&Epsilon;	GREEK CAPITAL LETTER EPSILON
Z	&#918;	&Zeta;	GREEK CAPITAL LETTER ZETA

Char	Number	Entity	Description
©	&#169;	&copy;	COPYRIGHT SIGN
®	&#174;	&reg;	REGISTERED SIGN
€	&#8364;	&euro;	EURO SIGN
™	&#8482;	&trade;	TRADEMARK
←	&#8592;	&larr;	LEFTWARDS ARROW
↑	&#8593;	&uarr;	UPWARDS ARROW
→	&#8594;	&rarr;	RIGHTWARDS ARROW
↓	&#8595;	&darr;	DOWNWARDS ARROW
♠	&#9824;	&spades;	BLACK SPADE SUIT
♣	&#9827;	&clubs;	BLACK CLUB SUIT
♥	&#9829;	&hearts;	BLACK HEART SUIT
♦	&#9830;	&diams;	BLACK DIAMOND SUIT



## 32. Using Emojis in HTML

Emojis are characters from the UTF-8 character set: 😊 😍 💕

### What are Emojis?

Emojis look like images, or icons, but they are not.

They are letters (characters) from the UTF-8 (Unicode) character set.

UTF-8 covers almost all of the characters and symbols in the world.

### The HTML charset Attribute

To display an HTML page correctly, a web browser must know the character set used in the page.

This is specified in the `<meta>` tag:

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

If not specified, UTF-8 is the default character set in HTML.

### UTF-8 Characters

Many UTF-8 characters cannot be typed on a keyboard, but they can always be displayed using numbers (called entity numbers):

- A is 65
- B is 66
- C is 67

## Example

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
</head>
<body>

<p>I will display A B C</p>
<p>I will display A B C</p>

</body>
</html>
```

## Example Explained

The `<meta charset="UTF-8">` element defines the character set.

The characters A, B, and C, are displayed by the numbers 65, 66, and 67.

To let the browser understand that you are displaying a character, you must start the entity number with `&#` and end it with `;` (semicolon).

## Emoji Characters

Emojis are also characters from the UTF-8 alphabet:












- 😊 is 128516
- 😍 is 128525
- 💕 is 128151

## Example

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
```



## Some Emoji Symbols in UTF-8

Emoji	Value
	&#128507;
	&#128508;
	&#128509;
	&#128510;
	&#128511;
	&#128512;
	&#128513;
	&#128514;
	&#128515;
	&#128516;
	&#128517;

## 33. HTML Encoding (Character Sets)

To display an HTML page correctly, a web browser must know which character set to use.

### From ASCII to UTF-8

ASCII was the first character encoding standard. ASCII defined 128 different characters that could be used on the internet: numbers (0-9), English letters (A-Z), and some special characters like ! \$ + - ( ) @ < > .

ISO-8859-1 was the default character set for HTML 4. This character set supported 256 different character codes. HTML 4 also supported UTF-8.

ANSI (Windows-1252) was the original Windows character set. ANSI is identical to ISO-8859-1, except that ANSI has 32 extra characters.

The HTML5 specification encourages web developers to use the UTF-8 character set, which covers almost all of the characters and symbols in the world!

### The HTML charset Attribute

To display an HTML page correctly, a web browser must know the character set used in the page.

This is specified in the `<meta>` tag:

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

## The ASCII Character Set

ASCII uses the values from 0 to 31 (and 127) for control characters.

ASCII uses the values from 32 to 126 for letters, digits, and symbols.

ASCII does not use the values from 128 to 255.

## The ANSI Character Set (Windows-1252)

ANSI is identical to ASCII for the values from 0 to 127.

ANSI has a proprietary set of characters for the values from 128 to 159.

ANSI is identical to UTF-8 for the values from 160 to 255.

## The ISO-8859-1 Character Set

ISO-8859-1 is identical to ASCII for the values from 0 to 127.

ISO-8859-1 does not use the values from 128 to 159.

ISO-8859-1 is identical to UTF-8 for the values from 160 to 255.

## The UTF-8 Character Set

UTF-8 is identical to ASCII for the values from 0 to 127. UTF-8 does not use the values from 128 to 159.

UTF-8 is identical to both ANSI and 8859-1 for the values from 160 to 255.

UTF-8 continues from the value 256 with more than 10 000 different characters.

## 34. HTML Uniform Resource Locators

A URL is another word for a web address.

A URL can be composed of words (e.g. w3schools.com), or an Internet Protocol (IP) address (e.g. 192.68.20.50).

Most people enter the name when surfing, because names are easier to remember than numbers.

### URL – Uniform Resource Locator

Web browsers request pages from web servers by using a URL.

A Uniform Resource Locator (URL) is used to address a document (or other data) on the web.

A web address like `https://www.sample.com/html/default.asp` follows these syntax rules:

***scheme://prefix.domain:port/path/filename***

#### Explanation:

- **scheme** - defines the **type** of Internet service (most common is **http** or **https**)
- **prefix** - defines a domain **prefix** (default for http is **www**)
- **domain** - defines the Internet **domain name** (like google.com)
- **port** - defines the **port number** at the host (default for http is **80**)
- **path** - defines a **path** at the server (If omitted: the root directory of the site)
- **filename** - defines the name of a document or resource

## Common URL Schemes

The table below lists some common schemes:

Scheme	Short for	Used for
http	HyperText Transfer Protocol	Common web pages. Not encrypted
https	Secure HyperText Transfer Protocol	Secure web pages. Encrypted
ftp	File Transfer Protocol	Downloading or uploading files
file		A file on your computer

## URL Encoding

URLs can only be sent over the Internet using the **ASCII character-set**. If a URL contains characters outside the ASCII set, the URL has to be converted.

URL encoding converts non-ASCII characters into a format that can be transmitted over the Internet.

URL encoding replaces non-ASCII characters with a "%" followed by hexadecimal digits.

URLs cannot contain spaces. URL encoding normally replaces a space with a plus (+) sign, or %20.

## Try It Yourself

Top of Form

Bottom of Form

If you click "Submit", the browser will URL encode the input before it is sent to the server.

A page at the server will display the received input.



Try some other input and click Submit again.

## ASCII Encoding Examples

Your browser will encode input, according to the character-set used in your page.

The default character-set in HTML5 is UTF-8.

Character	From Windows-1252	From UTF-8
€	%80	%E2%82%AC
£	%A3	%C2%A3
©	%A9	%C2%A9
®	%AE	%C2%AE
À	%C0	%C3%80
Á	%C1	%C3%81
Â	%C2	%C3%82
Ã	%C3	%C3%83
Ä	%C4	%C3%84
Å	%C5	%C3%85

## 35. HTML Versus XHTML

XHTML is a stricter, more XML-based version of HTML.

### What is XHTML?

- XHTML stands for **EX**tensible **HyperText Markup Language**
- XHTML is a stricter, more XML-based version of HTML
- XHTML is HTML defined as an XML application
- XHTML is supported by all major browsers

### Why XHTML?

XML is a markup language where all documents must be marked up correctly (be "well-formed").

XHTML was developed to make HTML more extensible and flexible to work with other data formats (such as XML). In addition, browsers ignore errors in HTML pages, and try to display the website even if it has some errors in the markup. So XHTML comes with a much stricter error handling.

If you want to study XML, please read our XML Tutorial.

### The Most Important Differences from HTML

- `<!DOCTYPE>` is **mandatory**
- The `xmlns` attribute in `<html>` is **mandatory**
- `<html>`, `<head>`, `<title>`, and `<body>` are **mandatory**
- Elements must always be **properly nested**
- Elements must always be **closed**
- Elements must always be in **lowercase**

- Attribute names must always be in **lowercase**
- Attribute values must always be **quoted**
- Attribute minimization is **forbidden**

## XHTML – <!DOCTYPE ....> Is Mandatory

An XHTML document must have an XHTML <!DOCTYPE> declaration.

The <html>, <head>, <title>, and <body> elements must also be present, and the xmlns attribute in <html> must specify the xml namespace for the document.

### Example

Here is an XHTML document with a minimum of required tags:

```
<!DOCTYPE html PUBLIC "-//sample//DTD XHTML 1.1//EN"
"http://www.sample.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.sample.org/1999/xhtml">
<head>
 <title>Title of document</title>
</head>
<body>

 some content here...

</body>
</html>
```

## XHTML Elements Must be Properly Nested

In XHTML, elements must always be properly nested within each other, like this:

### Correct:

```
<i>Some text</i>
```

## Wrong:

```
<i>Some text</i>
```

## XHTML Elements Must Always be Closed

In XHTML, elements must always be closed, like this:

## Correct:

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

## Wrong:

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

## XHTML Empty Elements Must Always be Closed

In XHTML, empty elements must always be closed, like this:

## Correct:

```
A break:

A horizontal rule: <hr />
An image:
```

## Wrong:

```
A break:

A horizontal rule: <hr>
An image:
```

## XHTML Elements Must be in Lowercase

In XHTML, element names must always be in lowercase, like this:

## Correct:

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

## Wrong:

```
<BODY>
<P>This is a paragraph</P>
</BODY>
```

## XHTML Attribute Names Must be in Lowercase

In XHTML, attribute names must always be in lowercase, like this:

## Correct:

```
Visit our HTML tutorial
```

## Wrong:

```
Visit our HTML tutorial
```

## XHTML Attribute Values Must be Quoted

In XHTML, attribute values must always be quoted, like this:

## Correct:

```
Visit our HTML tutorial
```

## Wrong:

```
Visit our HTML tutorial
```

## XHTML Attribute Minimization is Forbidden

In XHTML, attribute minimization is forbidden:

## Correct:

```
<input type="checkbox" name="vehicle" value="car" checked="checked" />
<input type="text" name="last name" disabled="disabled" />
```

## Wrong:

```
<input type="checkbox" name="vehicle" value="car" checked />
<input type="text" name="last name" disabled />
```