

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.
-

A Simple HTML Document

Example

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

<h1>My First Heading</h1>
<p>My first 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:

```
<tagname>Content goes here...</tagname>
```

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

[Learn HTML Using Notepad or TextEdit](#)

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 TextEdit (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 TextEdit.

[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 TextEdit \(Mac\)](#)

Open **Finder > Applications > TextEdit**

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

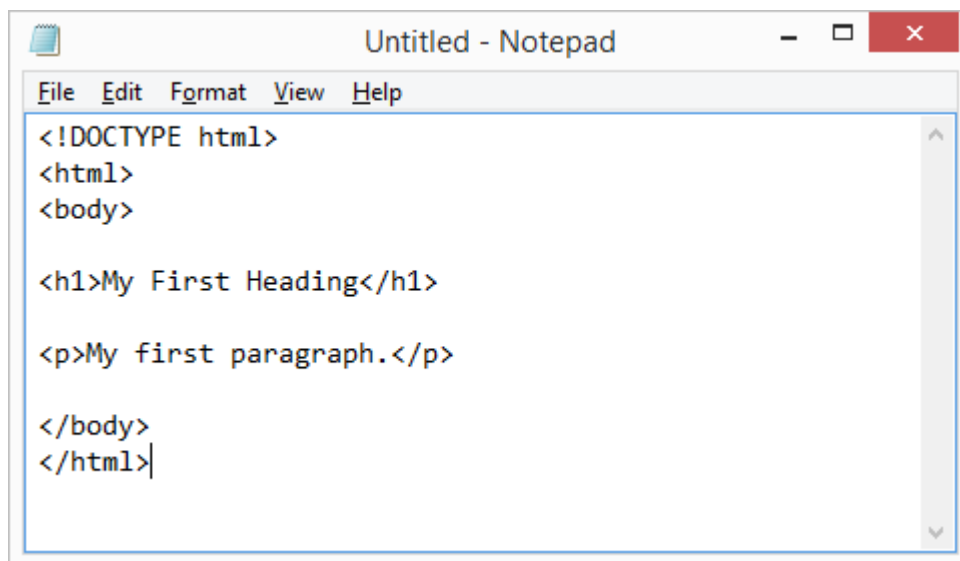
Write or copy the following HTML code into Notepad:

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

<h1>My First Heading</h1>

<p>My first paragraph.</p>

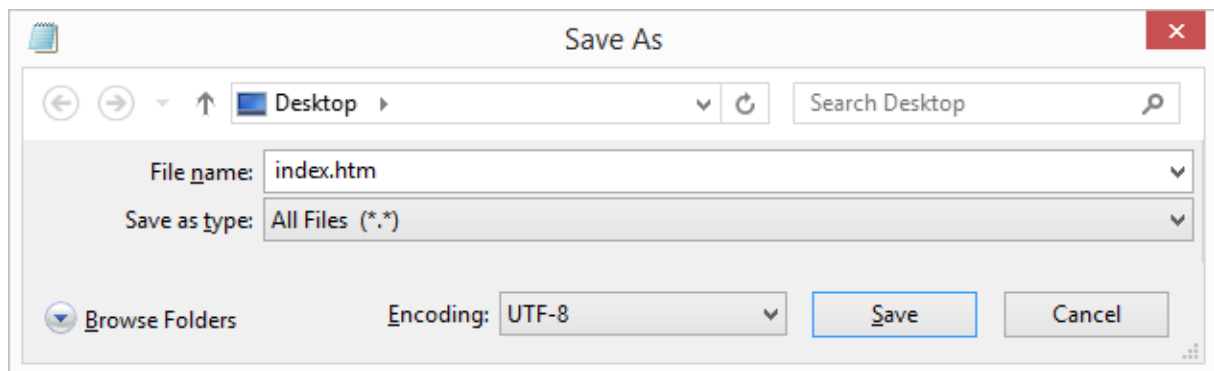
</body>
</html>
```



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).

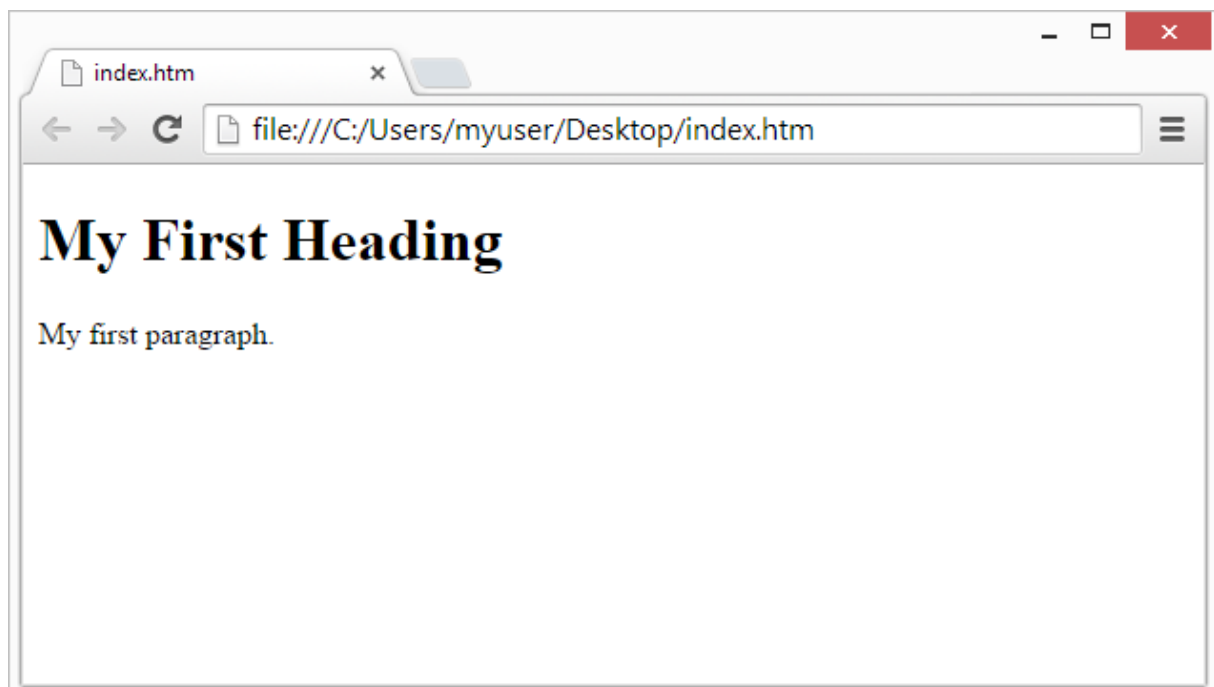


Tip: You can use either .htm or .html as file extension. There is no difference, it is up to you.

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:



HTML Basic Examples

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.w3schools.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.

HTML Elements

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

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

```
<tagname>Content goes here...</tagname>
```

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!

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 is 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 W3C **recommends** lowercase in HTML, and **demands** lowercase for stricter document types like XHTML.

HTML Tag Reference

GET THE COMPLETE HTML EXCEL REFERENCE

HTML Attributes

HTML attributes provide additional information about HTML elements.

- 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.w3schools.com">Visit W3Schools</a>
```

You will learn more about links in our [HTML Links chapter](#).

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

```

```

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.

Example

```

```

Example

See what happens if we try to display an image that does not exist:

```

```

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>
```

You will learn more about styles in our [HTML Styles chapter](#).

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, W3C **recommends** 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, W3C **recommends** quotes in HTML, and **demands** quotes for stricter document types like XHTML.

Good:

```
<a href="https://www.w3schools.com/html/">Visit our HTML tutorial</a>
```

Bad:

```
<a href=https://www.w3schools.com/html/>Visit our HTML tutorial</a>
```

Sometimes you have to use quotes. This example will not display the title attribute correctly, because it contains a space:

Example

```
<p title>About W3Schools>
```

Single or Double Quotes?

Double quotes around attribute values are the most common in HTML, but single quotes can also be used.

In some situations, when the attribute value itself contains double quotes, it is necessary to use single quotes:

```
p title='John "ShotGun" Nelson'>
```

Or vice versa:

```
<p title="John 'ShotGun' Nelson">
```

Chapter Summary

- All HTML elements can have **attributes**
- The `href` attribute of `<a>` specifies the URL of the page the link goes to
- The `src` attribute of `` specifies the path to the image to be displayed
- The `width` and `height` attributes of `` provide size information for images
- The `alt` attribute of `` provides an alternate text for an image
- The `style` attribute is used to add styles to an element, such as color, font, size, and more
- The `lang` attribute of the `<html>` tag declares the language of the Web page
- The `title` attribute defines some extra information about an element

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.w3schools.com">Visit W3Schools</a>
```

You will learn more about links in our [HTML Links chapter](#).

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:

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.

Example

```

```

Example

See what happens if we try to display an image that does not exist:

```

```

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>
```

HTML Paragraphs

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

HTML Paragraphs

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>
```

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>
```

Example

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>
```

Text Color

The CSS `color` property defines the text color for an HTML element:

Example

```
<h1 style="color:blue;">This is a heading</h1>
```

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

Fonts

The CSS `font-family` property defines the font to be used for an HTML element:

Example

```
<h1 style="font-family:verdana;">This is a heading</h1>
```

```
<p style="font-family:courier;">This is a paragraph.</p>
```

Text Size

The CSS `font-size` property defines the text size for an HTML element:

Example

```
<h1 style="font-size:300%;">This is a heading</h1>
<p style="font-size:160%;">This is a paragraph.</p>
```

Text Alignment

The CSS `text-align` property defines the horizontal text alignment for an HTML element:

Example

```
<h1 style="text-align:center;">Centered Heading</h1>
<p style="text-align:center;">Centered paragraph.</p>
```

Chapter Summary

- Use the `style` attribute for styling HTML elements
- Use `background-color` for background color
- Use `color` for text colors
- Use `font-family` for text fonts
- Use `font-size` for text sizes
- Use `text-align` for text alignment

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

```
<b>This text is bold</b>
```

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

Example

```
<strong>This text is important!</strong>
```

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

```
<em>This text is emphasized</em>
```

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 <sub>subscripted</sub> text.</p>
```

HTML <sup> Element

The HTML <sup> element defines superscript text. Superscript text appears half a character above the normal line, and is sometimes rendered in a smaller font. Superscript text can be used for footnotes, like WWW^[1]:

Example

```
<p>This is <sup>superscripted</sup> text.</p>
```

HTML Quotation and Citation Elements

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

Example

Here is a quote from WWF's website:

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 `<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.worldwildlife.org/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>
```

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 Exercises

HTML Quotation and Citation Elements

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

HTML Comments

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

HTML Comment Tag

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.

Add Comments

With comments you can place notifications and reminders in your HTML code:

Example

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

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

<!-- Remember to add more information here -->
```

Hide Content

Comments can be used to hide content.

Which can be helpful if you hide content temporarily:

Example

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

<!-- <p>This is another paragraph </p> -->

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

You can also hide more than one line, everything between the `<!--` and the `-->` will be hidden from the display.

Example

Hide a section of HTML code:

```
<p>This is a paragraph.</p>
<!--
<p>Look at this cool image:</p>
```

```

-->
<p>This is a paragraph too.</p>
```

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

Hide Inline Content

Comments can be used to hide parts in the middle of the HTML code.

Example

Hide a part of a paragraph:

```
<p>This <!-- great text --> is a paragraph.</p>
```

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:

Background Color

You can set the background color for HTML elements:

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 exerci tation 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:



Example

```
<h1 style="border:2px solid Tomato;">Hello World</h1>
<h1 style="border:2px solid DodgerBlue;">Hello World</h1>
<h1 style="border:2px solid Violet;">Hello World</h1>
```

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:

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.

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 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 how 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 property:

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

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.w3schools.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">
```

You can read more about file paths in the chapter [HTML File Paths](#).

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 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 W3Schools.com:

```
<a href="https://www.w3schools.com/">Visit W3Schools.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.w3schools.com/" target="_blank">Visit W3Schools!</a>
```

Example

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

```
<a href="https://www.w3schools.com/" target="_blank">Visit W3Schools!</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.w3.org/">W3C</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 `` 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):

Example

```
<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>
```

Tip: Learn more about JavaScript in our [JavaScript Tutorial](#).

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.w3schools.com/html/" title="Go to W3Schools HTML section">Visit our HTML Tutorial</a>
```

More on Absolute URLs and Relative URLs

Example

Use a full URL to link to a web page:

```
<a href="https://www.w3schools.com/html/default.asp">HTML tutorial</a>
```

Example

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

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

Example

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

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

You can read more about file paths in the chapter [HTML File Paths](#).

Chapter Summary

- Use the `<a>` element to define a link
- Use the `href` attribute to define the link address
- Use the `target` attribute to define where to open the linked document
- Use the `` element (inside `<a>`) to use an image as a link
- Use the `mailto:` scheme inside the `href` attribute to create a link that opens the user's email program

HTML Images

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

Example

```

```

Example

```

```

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 point to an image on another server.

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

Example

```

```

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>
```

Tip: To learn more about CSS Float, read our [CSS Float Tutorial](#).

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 `` 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

Note: Loading large images takes time, and can slow down your web page. Use images carefully.

HTML Favicon

A favicon is a small image displayed next to the page title in the browser tab.

How To Add a Favicon in HTML

You can use any image you like as your favicon. You can also create your own favicon on sites like <https://www.favicon.cc>.

Tip: A favicon is a small image, so it should be a simple image with high contrast.

A favicon image is displayed to the left of the page title in the browser tab, like this:

To add a favicon to your website, either save your favicon image to the root directory of your webserver, or create a folder in the root directory called images, and save your favicon image in this folder. A common name for a favicon image is "favicon.ico".

Next, add a `<link>` element to your "index.html" file, after the `<title>` element, like this:

Example

```
<!DOCTYPE html>
<html>
<head>
  <title>My Page Title</title>
  <link rel="icon" type="image/x-icon" href="/images/favicon.ico">
</head>
<body>

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

</body>
</html>
```

Now, save the "index.html" file and reload it in your browser. Your browser tab should now display your favicon image to the left of the page title.

Favicon File Format Support

The following table shows the file format support for a favicon image:

Browser	ICO	PNG	GIF	JPEG	SVG
Edge	Yes	Yes	Yes	Yes	Yes
Chrome	Yes	Yes	Yes	Yes	Yes
Firefox	Yes	Yes	Yes	Yes	Yes
Opera	Yes	Yes	Yes	Yes	Yes
Safari	Yes	Yes	Yes	Yes	Yes

Chapter Summary

- Use the HTML `<link>` element to insert a favicon

HTML Link Tag

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

[`<link>`](#)

Defines the relationship between a document and an external resource

For a complete list of all available HTML tags, visit our [HTML Tag Reference](#).

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

A table in HTML consists of table cells inside rows and columns

Example

A simple HTML table:

```
<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>
```

Table Cells

Each table cell is defined by a `<td>` and a `</td>` tag.

`td` stands for table data.

Everything between `<td>` and `</td>` are the content of the table cell.

Example

```
<table>
  <tr>
    <td>Emil</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
</table>
```

Note: table data elements are the data containers of the table. They can contain all sorts of HTML elements; text, images, lists, other tables, etc.

Table Rows

Each table row starts with a `<tr>` and end with a `</tr>` tag.

`tr` stands for table row.

Example

```
<table>
  <tr>
    <td>Emil</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
  <tr>
    <td>16</td>
    <td>14</td>
    <td>10</td>
  </tr>
</table>
```

You can have as many rows as you like in a table, just make sure that the number of cells are the same in each row.

Note: There are times where a row can have less or more cells than another. You will learn about that in a later chapter.

Table Headers

Sometimes you want your cells to be headers, in those cases use the `<th>` tag instead of the `<td>` tag:

Example

Let the first row be table headers:

```
<table>
  <tr>
    <th>Person 1</th>
    <th>Person 2</th>
    <th>Person 3</th>
  </tr>
  <tr>
    <td>Emil</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
  <tr>
    <td>16</td>
    <td>14</td>
    <td>10</td>
  </tr>
</table>
```

By default, the text in `<th>` elements are bold and centered, but you can change that with CSS.

Exercise:

Add a table row with two table headers.

The two table headers should have the value "Name" and "Age".

HTML Table Tags

Tag	Description
<code><table></code>	Defines a table
<code><th></code>	Defines a header cell in a table
<code><tr></code>	Defines a row in a table
<code><td></code>	Defines a cell in a table
<code><caption></code>	Defines a table caption
<code><colgroup></code>	Specifies a group of one or more columns in a table for formatting
<code><col></code>	Specifies column properties for each column within a <code><colgroup></code> element
<code><thead></code>	Groups the header content in a table
<code><tbody></code>	Groups the body content in a table
<code><tfoot></code>	Groups the footer content in a table

HTML Table Borders

HTML tables can have borders of different styles and shapes.

How To Add a Border

When you add a border to a table, you also add borders around each table cell:

To add a border, use the CSS `border` property on `table`, `th`, and `td` elements:

Example

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

Collapsed Table Borders

To avoid having double borders like in the example above, set the CSS `border-collapse` property to `collapse`.

This will make the borders collapse into a single border:

--	--	--

Example

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

Style Table Borders

If you set a background color of each cell, and give the border a white color (the same as the document background), you get the impression of an invisible border:

Example

```
table, th, td {  
  border: 1px solid white;  
  border-collapse: collapse;  
}  
th, td {  
  background-color: #96D4D4;  
}
```

[Try it Yourself »](#)

Round Table Borders

With the `border-radius` property, the borders get rounded corners:

--	--	--

Example

```
table, th, td {  
  border: 1px solid black;  
  border-radius: 10px;  
}
```

[Try it Yourself »](#)

Skip the border around the table by leaving out `table` from the css selector:

Example









```
th, td {  
  border: 1px solid black;  
  border-radius: 10px;  
}
```

[Try it Yourself »](#)

Dotted Table Borders

With the `border-style` property, you can set the appearance of the border.

The following values are allowed:

- dotted 
- dashed 
- solid 
- double 
- groove 
- ridge 
- inset 
- outset 
- none
- hidden

Example

```
th, td {
  border-style: dotted;
}
```

[Try it Yourself »](#)

Border Color

With the `border-color` property, you can set the color of the border.

Example

```
th, td {
  border-color: #96D4D4;
}
```

HTML Table Sizes

HTML tables can have different sizes for each column, row or the entire table.

Use the `style` attribute with the `width` or `height` properties to specify the size of a table, row or column.

HTML Table Width

To set the width of a table, add the `style` attribute to the `<table>` element:

Example

Set the width of the table to 100%:

```
<table style="width:100%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</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: Using a percentage as the size unit for a width means how wide will this element be compared to its parent element, which in this case is the `<body>` element.

HTML Table Column Width

--	--	--

To set the size of a specific column, add the `style` attribute on a `<th>` or `<td>` element:

Example

Set the width of the first column to 70%:

```
<table style="width:100%">
  <tr>
    <th style="width:70%">Firstname</th>
    <th>Lastname</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>
```

HTML Table Row Height

To set the height of a specific row, add the `style` attribute on a table row element:

Example

Set the height of the second row to 200 pixels:

```
<table style="width:100%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr style="height:200px">
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

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 `` tag. Each list item starts with the `` tag.

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

Example

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Ordered HTML List

An ordered list starts with the `` tag. Each list item starts with the `` tag.

The list items will be marked with numbers by default:

Example

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

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
<code></code>	Defines an unordered list
<code></code>	Defines an ordered list
<code></code>	Defines a list item
<code><dl></code>	Defines a description list
<code><dt></code>	Defines a term in a description list
<code><dd></code>	Describes the term in a description list

HTML Unordered Lists

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

Unordered HTML List

An unordered list starts with the `` tag. Each list item starts with the `` tag.

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

Example

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

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;">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Example - Circle

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

Example - Square

```
<ul style="list-style-type:square;">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Example - None

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

Nested HTML Lists

Lists can be nested (list inside list):

Example

```
<ul>
  <li>Coffee</li>
  <li>Tea
    <ul>
```

```
<li>Black tea</li>
<li>Green tea</li>
</ul>
</li>
<li>Milk</li>
</ul>
```

Note: A list item () 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>
```

```
<ul>
  <li><a href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>

</body>
</html>
```

Tip: You can learn much more about CSS in our [CSS Tutorial](#).

Chapter Summary

- Use the HTML `` element to define an unordered list
 - Use the CSS `list-style-type` property to define the list item marker
 - Use the HTML `` 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 `` tag defines an ordered list. An ordered list can be numerical or alphabetical.

Ordered HTML List

An ordered list starts with the `` tag. Each list item starts with the `` tag.

The list items will be marked with numbers by default:

Example

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

Ordered HTML List - The Type Attribute

The `type` attribute of the `` tag, defines the type of the list item marker:

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

Numbers:

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

Uppercase Letters:

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

Lowercase Letters:

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

Uppercase Roman Numbers:

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

Lowercase Roman Numbers:

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

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">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

Nested HTML Lists

Lists can be nested (list inside list):

Example

```
<ol>
  <li>Coffee</li>
  <li>Tea
    <ol>
      <li>Black tea</li>
      <li>Green tea</li>
    </ol>
  </li>
  <li>Milk</li>
</ol>
```

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

Chapter Summary

- Use the HTML `` element to define an ordered list
- Use the HTML `type` attribute to define the numbering type
- Use the HTML `` 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

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 the browsers automatically add some space (a margin) before and after the element.

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

Two commonly used block elements are: `<p>` and `<div>`.

The `<p>` element defines a paragraph in an HTML document.

The `<div>` element defines a division or a section in an HTML document.

The `<p>` element is a block-level element.

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

Example

`<p>Hello World</p>`

`<div>Hello World</div>`

Here are the block-level elements in HTML:

[<address>](#)

[<article>](#)

[<aside>](#)

[<blockquote>](#)

[<canvas>](#)

[<dd>](#)

[<div>](#)

[<dl>](#)

[<dt>](#)

[<fieldset>](#)

[<figcaption>](#)

[<figure>](#)

[<footer>](#)

[<form>](#)

[<h1>-<h6>](#)

[<header>](#)

[`<hr>`](#)

[``](#)

[`<main>`](#)

[`<nav>`](#)

[`<noscript>`](#)

[``](#)

[`<p>`](#)

[`<pre>`](#)

[`<section>`](#)

[`<table>`](#)

[`<tfoot>`](#)

[``](#)

[`<video>`](#)

Inline Elements

An inline element does not start on a new line.

An inline element only takes up as much width as necessary.

This is a `` element inside a paragraph.

Example

``Hello World``

Here are the inline elements in HTML:

[`<a>`](#)

[`<abbr>`](#)

[`<acronym>`](#)

[``](#)

[`<bdo>`](#)

[<big>](#)

[
](#)

[<button>](#)

[<cite>](#)

[<code>](#)

[<dfn>](#)

[](#)

[<i>](#)

[](#)

[<input>](#)

[<kbd>](#)

[<label>](#)

[<map>](#)

[<object>](#)

[<output>](#)

[<q>](#)

[<samp>](#)

[<script>](#)

[<select>](#)

[<small>](#)

[](#)

[](#)

[<sub>](#)

[<sup>](#)

[<textarea>](#)

[<time>](#)

[<tt>](#)

`<var>`

Note: An inline element cannot contain a block-level element!

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 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 `` Element

The `` element is an inline container used to mark up a part of a text, or a part of a document.

The `` element has no required attributes, but `style`, `class` and `id` are common.

When used together with CSS, the `` element can be used to style parts of the text:

Example

```
<p>My mother has <span style="color:blue;font-weight:bold">blue</span> eyes and my father has
<span style="color:darkolivegreen;font-weight:bold">dark green</span> 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 `` element is an inline container used to mark up a part of a text, or a part of a document
-

HTML Tags

Tag	Description
<code><div></code>	Defines a section in a document (block-level)
<code></code>	Defines a section in a document (inline)

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 `` elements with a `class` attribute with the value of "note". Both `` 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 <span class="note">Important</span> Heading</h1>
<p>This is some <span class="note">important</span> text.</p>

</body>
</html>
```

Tip: The `class` attribute can be used on **any** HTML element.

Note: The class name is case sensitive!

Tip: You can learn much more about CSS in our [CSS Tutorial](#).

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>
```

Don't worry if you don't understand the code in the example above.

You will learn more about JavaScript in our [HTML JavaScript](#) chapter, or you can study our [JavaScript Tutorial](#).

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

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, cannot start with a number, 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>
```

Tip: You can learn much more about CSS in our [CSS Tutorial](#).

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:

Example

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

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

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

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:

Example

Use the `id` attribute to manipulate text with JavaScript:

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

Tip: Study JavaScript in the [HTML JavaScript](#) chapter, or in our [JavaScript Tutorial](#).

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

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"></iframe>
```

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:

Example

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

```
<p><a href="https://www.w3schools.com" target="iframe_a">W3Schools.com</a></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

HTML iframe Tag

Tag	Description
<code><iframe></code>	Defines an inline frame

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">
```

Tip: To learn all about CSS, visit our [CSS Tutorial](#).

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.



[Without the viewport meta tag](#)



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. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Nam liber tempor cum soluta nobis eleifend congue nihil imperdiet domine...

[With the viewport meta tag](#)

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>
```

Tip: To learn all about JavaScript, visit our [JavaScript Tutorial](#).

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.w3schools.com/" target="_blank">
</head>

<body>

<a href="tags/tag_base.asp">HTML base Tag</a>
</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><head></code>	Defines information about the document
<code><title></code>	Defines the title of a document
<code><base></code>	Defines a default address or a default target for all links on a page
<code><link></code>	Defines the relationship between a document and an external resource
<code><meta></code>	Defines metadata about an HTML document
<code><script></code>	Defines a client-side script
<code><style></code>	Defines style information for a document

HTML Layout Elements and Techniques

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

Example

Cities

- [London](#)
- [Paris](#)
- [Tokyo](#)

London

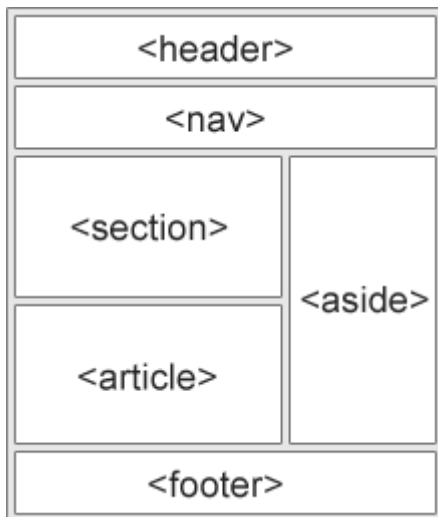
London is the capital city of England. It is the most populous city in the United Kingdom, with a metropolitan area of over 13 million inhabitants.

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

You can read more about semantic elements in our [HTML Semantics](#) chapter.

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 Frameworks

If you want to create your layout fast, you can use a CSS framework, like [W3.CSS](#) or [Bootstrap](#).

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. Learn more about float in our [CSS Float and Clear](#) chapter.

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

CSS Flexbox Layout

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

Learn more about flexbox in our [CSS Flexbox](#) chapter.

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.

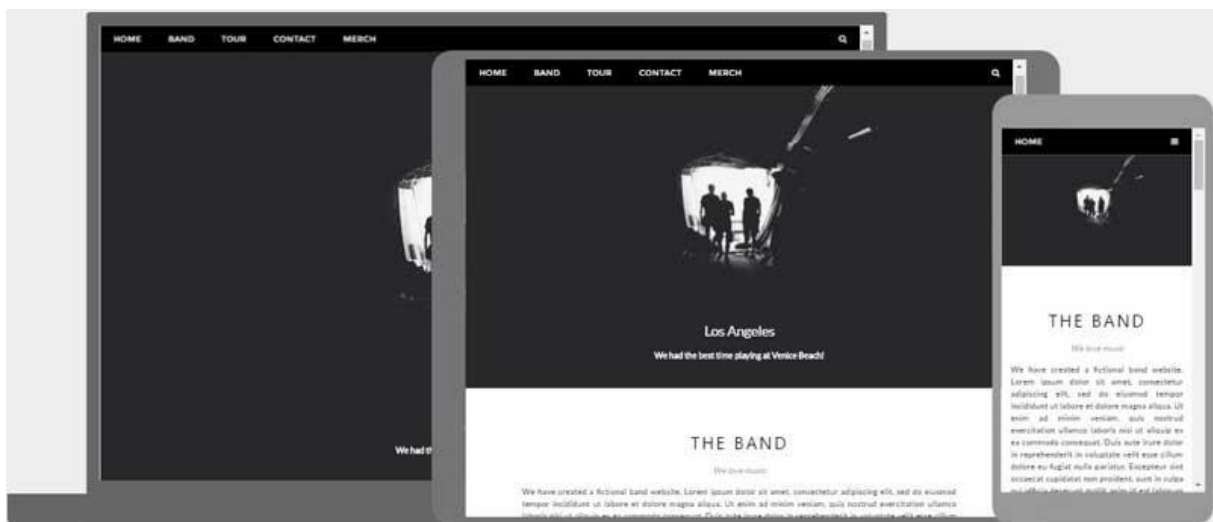
Standing on the River Thames, London has been a major settlement for two millennia, its history going back to its founding by the Romans, who named it Londinium.

Footer

HTML 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:

Without the viewport meta tag:



With the viewport meta tag:



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
suscepit lobortis nisl ut aliquip ex ea commodo
consequat. Duis autem vel eum iriure dolor in
hendrerit in vulputate velit esse molestie
consequat, vel illum dolore eu feugiat nulla
facilisis at vero eros et accumsan et iusto odio
dignissim qui blandit praesent luptatum zzril
delenit augue duis dolore te feugait nulla
facilisi. Nam liber tempor cum soluta nobis
eleifend antian congue nihil imperdiet domine...

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

ADVERTISEMENT

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

```

```

Show Different Images Depending on Browser Width

The HTML `<picture>` element allows you to define different images for different browser window sizes.

Resize the browser window to see how the image below change depending on the width:



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.

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: **<** or **<**;

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: ** **;

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 ** ** character entity.

Tip: The non-breaking hyphen ([‑](#)) is used to define a hyphen character (-) that does not break into a new line.

ADVERTISEMENT

Some Useful HTML Character Entities

Result	Description	Entity Name	Entity Number	Try it
	non-breaking space	 	 	Try it »
<	less than	<	<	Try it »
>	greater than	>	>	Try it »
&	ampersand	&	&	Try it »
"	double quotation mark	"	"	Try it »
'	single quotation mark (apostrophe)	'	'	Try it »
¢	cent	¢	¢	Try it »

£	pound	£	£	Try it »
¥	yen	¥	¥	Try it »
€	euro	€	€	Try it »
©	copyright	©	©	Try it »
®	registered trademark	®	®	Try it »

Note: 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	Try it
`	a	à	à	Try it »

'	a	á	á	Try it »
^	a	â	â	Try it »
~	a	ã	ã	Try it »
`	o	ò	ò	Try it »
'	o	ó	ó	Try it »
^	o	ô	ô	Try it »
~	o	õ	õ	Try it »

You will see more HTML symbols in the next chapter of this tutorial.

NEXT TOPIC: HTML FORMS

HTML Forms

An HTML form is used to collect user input. The user input can then be sent to a server for processing.

The <form> Element

The HTML <form> element defines a form that is used to collect user input:

```
<form>
.
form elements
.
</form>
```

An HTML form contains **form elements**.

Form elements are different types of input elements, like: text fields, checkboxes, radio buttons, submit buttons, and more.

The <input> Element

The <input> element is the most important form element.

The <input> element is displayed in several ways, depending on the **type** attribute.

Here are some examples:

Type	Description
<input type="text">	Defines a single-line text input field
<input type="radio">	Defines a radio button (for selecting one of many choices)
<input type="submit">	Defines a submit button (for submitting the form)

You will learn a lot more about input types later in this tutorial.

Text Fields

<input type="text"> defines a single-line input field for **text input**.

Example

A form with two text input fields:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

Last name:

Note: The form itself is not visible. Also note that the default width of an input field is 20 characters.

The <label> Element

Notice the use of the <label> element in the example above.

The <label> tag defines a label for many form elements.

The <label> element is useful for screen-reader users, because the screen-reader will read out loud the label when the user focus on the input element.

The <label> element also help users who have difficulty clicking on very small regions (such as radio buttons or checkboxes) - because when the user clicks the text within the <label> element, it toggles the radio button/checkbox.

The `for` attribute of the <label> tag should be equal to the `id` attribute of the <input> element to bind them together.

Radio Buttons

The <input type="radio"> defines a radio button.

Radio buttons let a user select ONE of a limited number of choices.

Example

A form with radio buttons:

<p>Choose your favorite Web language:</p>

```
<form>
  <input type="radio" id="html" name="fav_language" value="HTML">
  <label for="html">HTML</label><br>
  <input type="radio" id="css" name="fav_language" value="CSS">
  <label for="css">CSS</label><br>
  <input type="radio" id="javascript" name="fav_language" value="JavaScript">
  <label for="javascript">JavaScript</label>
</form>
```

This is how the HTML code above will be displayed in a browser:

Choose your favorite Web language:

- ☐ HTML
 - ☐ CSS
 - ☐ JavaScript
-

Checkboxes

The <input type="checkbox"> defines a **checkbox**.

Checkboxes let a user select ZERO or MORE options of a limited number of choices.

Example

A form with checkboxes:

```
<form>
  <input type="checkbox" id="vehicle1" name="vehicle1" value="Bike">
  <label for="vehicle1"> I have a bike</label><br>
  <input type="checkbox" id="vehicle2" name="vehicle2" value="Car">
  <label for="vehicle2"> I have a car</label><br>
  <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat">
  <label for="vehicle3"> I have a boat</label>
</form>
```

This is how the HTML code above will be displayed in a browser:

- ☐ I have a bike
 - ☐ I have a car
 - ☐ I have a boat
-

The Submit Button

The `<input type="submit">` defines a button for submitting the form data to a form-handler.

The form-handler is typically a file on the server with a script for processing input data.

The form-handler is specified in the form's `action` attribute.

Example

A form with a submit button:

```
<form action="/action_page.php">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>
  <input type="submit" value="Submit">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

Last name:

The Name Attribute for `<input>`

Notice that each input field must have a `name` attribute to be submitted.

If the `name` attribute is omitted, the value of the input field will not be sent at all.

Example

This example will not submit the value of the "First name" input field:

```
<form action="/action_page.php">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" value="John"><br><br>
  <input type="submit" value="Submit">
</form>
```

HTML Form Attributes

This chapter describes the different attributes for the HTML `<form>` element.

The Action Attribute

The `action` attribute defines the action to be performed when the form is submitted.

Usually, the form data is sent to a file on the server when the user clicks on the submit button.

In the example below, the form data is sent to a file called "action_page.php". This file contains a server-side script that handles the form data:

Example

On submit, send form data to "action_page.php":

```
<form action="/action_page.php">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>
  <input type="submit" value="Submit">
</form>
```

Tip: If the `action` attribute is omitted, the action is set to the current page.

The Target Attribute

The `target` attribute specifies where to display the response that is received after submitting the form.

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

Value	Description
<code>_blank</code>	The response is displayed in a new window or tab
<code>_self</code>	The response is displayed in the current window
<code>_parent</code>	The response is displayed in the parent frame
<code>_top</code>	The response is displayed in the full body of the window

framename The response is displayed in a named iframe

The default value is `_self` which means that the response will open in the current window.

Example

Here, the submitted result will open in a new browser tab:

```
<form action="/action_page.php" target="_blank">
```

The Method Attribute

The `method` attribute specifies the HTTP method to be used when submitting the form data.

The form-data can be sent as URL variables (with `method="get"`) or as HTTP post transaction (with `method="post"`).

The default HTTP method when submitting form data is GET.

Example

This example uses the GET method when submitting the form data:

```
<form action="/action_page.php" method="get">
```

Example

This example uses the POST method when submitting the form data:

```
<form action="/action_page.php" method="post">
```

Notes on GET:

- Appends the form data to the URL, in name/value pairs
- NEVER use GET to send sensitive data! (the submitted form data is visible in the URL!)
- The length of a URL is limited (2048 characters)
- Useful for form submissions where a user wants to bookmark the result
- GET is good for non-secure data, like query strings in Google

Notes on POST:

- Appends the form data inside the body of the HTTP request (the submitted form data is not shown in the URL)
- POST has no size limitations, and can be used to send large amounts of data.
- Form submissions with POST cannot be bookmarked

Tip: Always use POST if the form data contains sensitive or personal information!

The Autocomplete Attribute

The `autocomplete` attribute specifies whether a form should have autocomplete on or off.

When autocomplete is on, the browser automatically complete values based on values that the user has entered before.

Example

A form with autocomplete on:

```
<form action="/action_page.php" autocomplete="on">
```

The Novalidate Attribute

The `novalidate` attribute is a boolean attribute.

When present, it specifies that the form-data (input) should not be validated when submitted.

Example

A form with a novalidate attribute:

```
<form action="/action_page.php" novalidate>
```

HTML Form Elements

This chapter describes all the different HTML form elements.

The HTML `<form>` Elements

The HTML `<form>` element can contain one or more of the following form elements:

- `<input>`
- `<label>`
- `<select>`
- `<textarea>`

- `<button>`
- `<fieldset>`
- `<legend>`
- `<datalist>`
- `<output>`
- `<option>`
- `<optgroup>`

The `<input>` Element

One of the most used form element is the `<input>` element.

The `<input>` element can be displayed in several ways, depending on the `type` attribute.

Example

```
<label for="fname">First name:</label>
<input type="text" id="fname" name="fname">
```

All the different values of the `type` attribute are covered in the next chapter: [HTML Input Types](#).

The `<label>` Element

The `<label>` element defines a label for several form elements.

The `<label>` element is useful for screen-reader users, because the screen-reader will read out loud the label when the user focus on the input element.

The `<label>` element also help users who have difficulty clicking on very small regions (such as radio buttons or checkboxes) - because when the user clicks the text within the `<label>` element, it toggles the radio button/checkbox.

The `for` attribute of the `<label>` tag should be equal to the `id` attribute of the `<input>` element to bind them together.

The `<select>` Element

The `<select>` element defines a drop-down list:

Example

```
<label for="cars">Choose a car:</label>
<select id="cars" name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
```

```
<option value="audi">Audi</option>
</select>
```

The `<option>` element defines an option that can be selected.

By default, the first item in the drop-down list is selected.

To define a pre-selected option, add the `selected` attribute to the option:

Example

```
<option value="fiat" selected>Fiat</option>
```

Visible Values:

Use the `size` attribute to specify the number of visible values:

Example

```
<label for="cars">Choose a car:</label>
<select id="cars" name="cars" size="3">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

Allow Multiple Selections:

Use the `multiple` attribute to allow the user to select more than one value:

Example

```
<label for="cars">Choose a car:</label>
<select id="cars" name="cars" size="4" multiple>
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

The `<textarea>` Element

The `<textarea>` element defines a multi-line input field (a text area):

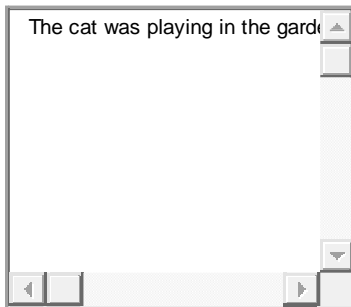
Example

```
<textarea name="message" rows="10" cols="30">
The cat was playing in the garden.
</textarea>
```

The `rows` attribute specifies the visible number of lines in a text area.

The `cols` attribute specifies the visible width of a text area.

This is how the HTML code above will be displayed in a browser:



You can also define the size of the text area by using CSS:

Example

```
<textarea name="message" style="width:200px; height:600px;">
```

The cat was playing in the garden.

```
</textarea>
```

The <button> Element

The `<button>` element defines a clickable button:

Example

```
<button type="button" onclick="alert('Hello World!')">Click Me!</button>
```

This is how the HTML code above will be displayed in a browser:

Note: Always specify the `type` attribute for the button element. Different browsers may use different default types for the button element.

The <fieldset> and <legend> Elements

The `<fieldset>` element is used to group related data in a form.

The `<legend>` element defines a caption for the `<fieldset>` element.

Example

```
<form action="/action_page.php">
```

```
<fieldset>
```

```
<legend>Personalia:</legend>
```

```
<label for="fname">First name:</label><br>
<input type="text" id="fname" name="fname" value="John"><br>
<label for="lname">Last name:</label><br>
<input type="text" id="lname" name="lname" value="Doe"><br><br>
<input type="submit" value="Submit">
</fieldset>
</form>
```

This is how the HTML code above will be displayed in a browser:

Personalia: First name:

Last name:

The <datalist> Element

The <datalist> element specifies a list of pre-defined options for an <input> element.

Users will see a drop-down list of the pre-defined options as they input data.

The list attribute of the <input> element, must refer to the id attribute of the <datalist> element.

Example

```
<form action="/action_page.php">
  <input list="browsers">
  <datalist id="browsers">
    <option value="Internet Explorer">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
</form>
```

The <output> Element

The <output> element represents the result of a calculation (like one performed by a script).

Example

Perform a calculation and show the result in an <output> element:

```
<form action="/action_page.php"
oninput="x.value=parseInt(a.value)+parseInt(b.value)">
0
<input type="range" id="a" name="a" value="50">
100 +
<input type="number" id="b" name="b" value="50">
=
<output name="x" for="a b"></output>
<br><br>
<input type="submit">
</form>
```

HTML Input Types

This chapter describes the different types for the HTML `<input>` element.

HTML Input Types

Here are the different input types you can use in HTML:

- `<input type="button">`
- `<input type="checkbox">`
- `<input type="color">`
- `<input type="date">`
- `<input type="datetime-local">`
- `<input type="email">`
- `<input type="file">`
- `<input type="hidden">`
- `<input type="image">`
- `<input type="month">`
- `<input type="number">`
- `<input type="password">`
- `<input type="radio">`
- `<input type="range">`
- `<input type="reset">`
- `<input type="search">`
- `<input type="submit">`
- `<input type="tel">`
- `<input type="text">`

- `<input type="time">`
- `<input type="url">`
- `<input type="week">`

Tip: The default value of the `type` attribute is "text".

Input Type Text

`<input type="text">` defines a **single-line text input field**:

Example

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

Last name:

Input Type Password

`<input type="password">` defines a **password field**:

Example

```
<form>
  <label for="username">Username:</label><br>
  <input type="text" id="username" name="username"><br>
  <label for="pwd">Password:</label><br>
  <input type="password" id="pwd" name="pwd">
</form>
```

This is how the HTML code above will be displayed in a browser:

Username:

Password:

The characters in a password field are masked (shown as asterisks or circles).

Input Type Submit

`<input type="submit">` defines a button for **submitting** form data to a **form-handler**.

The form-handler is typically a server page with a script for processing input data.

The form-handler is specified in the form's `action` attribute:

Example

```
<form action="/action_page.php">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>
  <input type="submit" value="Submit">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

Last name:

If you omit the submit button's value attribute, the button will get a default text:

Example

```
<form action="/action_page.php">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>
  <input type="submit">
</form>
```

Input Type Reset

`<input type="reset">` defines a **reset button** that will reset all form values to their default values:

Example

```
<form action="/action_page.php">
  <label for="fname">First name:</label><br>
```



```
<input type="text" id="fname" name="fname" value="John"><br>
<label for="lname">Last name:</label><br>
<input type="text" id="lname" name="lname" value="Doe"><br><br>
<input type="submit" value="Submit">
<input type="reset">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

Last name:

If you change the input values and then click the "Reset" button, the form-data will be reset to the default values.

Input Type Radio

`<input type="radio">` defines a **radio button**.

Radio buttons let a user select ONLY ONE of a limited number of choices:

Example

`<p>Choose your favorite Web language:</p>`

```
<form>
  <input type="radio" id="html" name="fav_language" value="HTML">
  <label for="html">HTML</label><br>
  <input type="radio" id="css" name="fav_language" value="CSS">
  <label for="css">CSS</label><br>
  <input type="radio" id="javascript" name="fav_language" value="JavaScript">
  <label for="javascript">JavaScript</label>
</form>
```

This is how the HTML code above will be displayed in a browser:

- ☐ HTML
- ☐ CSS
- ☐ JavaScript

Input Type Checkbox

`<input type="checkbox">` defines a **checkbox**.

Checkboxes let a user select ZERO or MORE options of a limited number of choices.

Example

```
<form>
  <input type="checkbox" id="vehicle1" name="vehicle1" value="Bike">
  <label for="vehicle1"> I have a bike</label><br>
  <input type="checkbox" id="vehicle2" name="vehicle2" value="Car">
  <label for="vehicle2"> I have a car</label><br>
  <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat">
  <label for="vehicle3"> I have a boat</label>
</form>
```

This is how the HTML code above will be displayed in a browser:

- ☐ I have a bike
 - ☐ I have a car
 - ☐ I have a boat
-

Input Type Button

`<input type="button">` defines a **button**:

Example

```
<input type="button" onclick="alert('Hello World!')" value="Click Me!">
```

This is how the HTML code above will be displayed in a browser:

Input Type Color

The `<input type="color">` is used for input fields that should contain a color.

Depending on browser support, a color picker can show up in the input field.

Example

```
<form>
  <label for="favcolor">Select your favorite color:</label>
  <input type="color" id="favcolor" name="favcolor">
</form>
```

Input Type Date

The `<input type="date">` is used for input fields that should contain a date.

Depending on browser support, a date picker can show up in the input field.

Example

```
<form>
  <label for="birthday">Birthday:</label>
  <input type="date" id="birthday" name="birthday">
</form>
```

You can also use the `min` and `max` attributes to add restrictions to dates:

Example

```
<form>
  <label for="datemax">Enter a date before 1980-01-01:</label>
  <input type="date" id="datemax" name="datemax" max="1979-12-31"><br><br>
  <label for="datemin">Enter a date after 2000-01-01:</label>
  <input type="date" id="datemin" name="datemin" min="2000-01-02">
</form>
```

Input Type Datetime-local

The `<input type="datetime-local">` specifies a date and time input field, with no time zone.

Depending on browser support, a date picker can show up in the input field.

Example

```
<form>
  <label for="birthdaytime">Birthday (date and time):</label>
  <input type="datetime-local" id="birthdaytime" name="birthdaytime">
</form>
```

Input Type Email

The `<input type="email">` is used for input fields that should contain an e-mail address.

Depending on browser support, the e-mail address can be automatically validated when submitted.

Some smartphones recognize the email type, and add ".com" to the keyboard to match email input.

Example

```
<form>
  <label for="email">Enter your email:</label>
  <input type="email" id="email" name="email">
</form>
```

Input Type File

The `<input type="file">` defines a file-select field and a "Browse" button for file uploads.

Example

```
<form>
  <label for="myfile">Select a file:</label>
  <input type="file" id="myfile" name="myfile">
</form>
```

Input Type Hidden

The `<input type="hidden">` defines a hidden input field (not visible to a user).

A hidden field lets web developers include data that cannot be seen or modified by users when a form is submitted.

A hidden field often stores what database record that needs to be updated when the form is submitted.

Note: While the value is not displayed to the user in the page's content, it is visible (and can be edited) using any browser's developer tools or "View Source" functionality. Do not use hidden inputs as a form of security!

Example

```
<form>
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <input type="hidden" id="custId" name="custId" value="3487">
  <input type="submit" value="Submit">
</form>
```

Input Type Month

The `<input type="month">` allows the user to select a month and year.

Depending on browser support, a date picker can show up in the input field.

Example

```
<form>
  <label for="bdaymonth">Birthday (month and year):</label>
  <input type="month" id="bdaymonth" name="bdaymonth">
</form>
```

Input Type Number

The `<input type="number">` defines a **numeric** input field.

You can also set restrictions on what numbers are accepted.

The following example displays a numeric input field, where you can enter a value from 1 to 5:

Example

```
<form>
  <label for="quantity">Quantity (between 1 and 5):</label>
  <input type="number" id="quantity" name="quantity" min="1" max="5">
</form>
```

Input Restrictions

Here is a list of some common input restrictions:

Attribute	Description
checked	Specifies that an input field should be pre-selected when the page loads (for type="checkbox" or type="radio")
disabled	Specifies that an input field should be disabled
max	Specifies the maximum value for an input field
maxlength	Specifies the maximum number of character for an input field
min	Specifies the minimum value for an input field
pattern	Specifies a regular expression to check the input value against
readonly	Specifies that an input field is read only (cannot be changed)
required	Specifies that an input field is required (must be filled out)
size	Specifies the width (in characters) of an input field
step	Specifies the legal number intervals for an input field
value	Specifies the default value for an input field

You will learn more about input restrictions in the next chapter.

The following example displays a numeric input field, where you can enter a value from 0 to 100, in steps of 10. The default value is 30:

Example

```
<form>
  <label for="quantity">Quantity:</label>
  <input type="number" id="quantity" name="quantity" min="0" max="100" step="10" value="30">
</form>
```

Input Type Range

The `<input type="range">` defines a control for entering a number whose exact value is not important (like a slider control). Default range is 0 to 100. However, you can set restrictions on what numbers are accepted with the `min`, `max`, and `step` attributes:

Example

```
<form>
  <label for="vol">Volume (between 0 and 50):</label>
  <input type="range" id="vol" name="vol" min="0" max="50">
</form>
```

Input Type Search

The `<input type="search">` is used for search fields (a search field behaves like a regular text field).

Example

```
<form>
  <label for="gsearch">Search Google:</label>
  <input type="search" id="gsearch" name="gsearch">
</form>
```

Input Type Tel

The `<input type="tel">` is used for input fields that should contain a telephone number.

Example

```
<form>
  <label for="phone">Enter your phone number:</label>
  <input type="tel" id="phone" name="phone" pattern="[0-9]{3}-[0-9]{2}-[0-9]{3}">
</form>
```

Input Type Time

The `<input type="time">` allows the user to select a time (no time zone).

Depending on browser support, a time picker can show up in the input field.

Example

```
<form>
  <label for="appt">Select a time:</label>
  <input type="time" id="appt" name="appt">
</form>
```

Input Type Url

The `<input type="url">` is used for input fields that should contain a URL address.

Depending on browser support, the url field can be automatically validated when submitted.

Some smartphones recognize the url type, and adds ".com" to the keyboard to match url input.

Example

```
<form>
  <label for="homepage">Add your homepage:</label>
  <input type="url" id="homepage" name="homepage">
</form>
```

Input Type Week

The `<input type="week">` allows the user to select a week and year.

Depending on browser support, a date picker can show up in the input field.

Example

```
<form>
  <label for="week">Select a week:</label>
  <input type="week" id="week" name="week">
</form>
```

HTML Input Attributes

This chapter describes the different attributes for the HTML `<input>` element.

The value Attribute

The input `value` attribute specifies an initial value for an input field:

Example

Input fields with initial (default) values:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe">
</form>
```

The readonly Attribute

The input `readonly` attribute specifies that an input field is read-only.

A read-only input field cannot be modified (however, a user can tab to it, highlight it, and copy the text from it).

The value of a read-only input field will be sent when submitting the form!

Example

A read-only input field:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John" readonly><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe">
</form>
```

The disabled Attribute

The input `disabled` attribute specifies that an input field should be disabled.

A disabled input field is unusable and un-clickable.

The value of a disabled input field will not be sent when submitting the form!

Example

A disabled input field:


```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John" disabled><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe">
</form>
```

ADVERTISEMENT

The size Attribute

The input `size` attribute specifies the visible width, in characters, of an input field.

The default value for `size` is 20.

Note: The `size` attribute works with the following input types: text, search, tel, url, email, and password.

Example

Set a width for an input field:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" size="50"><br>
  <label for="pin">PIN:</label><br>
  <input type="text" id="pin" name="pin" size="4">
</form>
```

The maxlength Attribute

The input `maxlength` attribute specifies the maximum number of characters allowed in an input field.

Note: When a `maxlength` is set, the input field will not accept more than the specified number of characters. However, this attribute does not provide any feedback. So, if you want to alert the user, you must write JavaScript code.

Example

Set a maximum length for an input field:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" size="50"><br>
  <label for="pin">PIN:</label><br>
```

```
<input type="text" id="pin" name="pin" maxlength="4" size="4">
</form>
```

The min and max Attributes

The `input min` and `max` attributes specify the minimum and maximum values for an input field.

The `min` and `max` attributes work with the following input types: number, range, date, datetime-local, month, time and week.

Tip: Use the `max` and `min` attributes together to create a range of legal values.

Example

Set a max date, a min date, and a range of legal values:

```
<form>
  <label for="datemax">Enter a date before 1980-01-01:</label>
  <input type="date" id="datemax" name="datemax" max="1979-12-31"><br><br>

  <label for="datemin">Enter a date after 2000-01-01:</label>
  <input type="date" id="datemin" name="datemin" min="2000-01-02"><br><br>

  <label for="quantity">Quantity (between 1 and 5):</label>
  <input type="number" id="quantity" name="quantity" min="1" max="5">
</form>
```

The multiple Attribute

The `input multiple` attribute specifies that the user is allowed to enter more than one value in an input field.

The `multiple` attribute works with the following input types: email, and file.

Example

A file upload field that accepts multiple values:

```
<form>
  <label for="files">Select files:</label>
  <input type="file" id="files" name="files" multiple>
</form>
```

The pattern Attribute

The input `pattern` attribute specifies a regular expression that the input field's value is checked against, when the form is submitted.

The `pattern` attribute works with the following input types: text, date, search, url, tel, email, and password.

Tip: Use the global [title](#) attribute to describe the pattern to help the user.

Tip: Learn more about [regular expressions](#) in our JavaScript tutorial.

Example

An input field that can contain only three letters (no numbers or special characters):

```
<form>
  <label for="country_code">Country code:</label>
  <input type="text" id="country_code" name="country_code"
    pattern="[A-Za-z]{3}" title="Three letter country code">
</form>
```

The placeholder Attribute

The input `placeholder` attribute specifies a short hint that describes the expected value of an input field (a sample value or a short description of the expected format).

The short hint is displayed in the input field before the user enters a value.

The `placeholder` attribute works with the following input types: text, search, url, tel, email, and password.

Example

An input field with a placeholder text:

```
<form>
  <label for="phone">Enter a phone number:</label>
  <input type="tel" id="phone" name="phone"
    placeholder="123-45-678"
    pattern="[0-9]{3}-[0-9]{2}-[0-9]{3}">
</form>
```

The required Attribute

The input `required` attribute specifies that an input field must be filled out before submitting the form.

The `required` attribute works with the following input types: text, search, url, tel, email, password, date pickers, number, checkbox, radio, and file.

Example

A required input field:

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

The step Attribute

The input `step` attribute specifies the legal number intervals for an input field.

Example: if `step="3"`, legal numbers could be -3, 0, 3, 6, etc.

Tip: This attribute can be used together with the `max` and `min` attributes to create a range of legal values.

The `step` attribute works with the following input types: number, range, date, datetime-local, month, time and week.

Example

An input field with a specified legal number intervals:

```
<form>
  <label for="points">Points:</label>
  <input type="number" id="points" name="points" step="3">
</form>
```

Note: Input restrictions are not foolproof, and JavaScript provides many ways to add illegal input. To safely restrict input, it must also be checked by the receiver (the server)!

The autofocus Attribute

The input `autofocus` attribute specifies that an input field should automatically get focus when the page loads.

Example

Let the "First name" input field automatically get focus when the page loads:

```
<form>
  <label for="fname">First name:</label><br>
```

```
<input type="text" id="fname" name="fname" autofocus><br>
<label for="lname">Last name:</label><br>
<input type="text" id="lname" name="lname">
</form>
```

The height and width Attributes

The `input` `height` and `width` attributes specify the height and width of an `<input type="image">` element.

Tip: Always specify both the height and width attributes for images. If height and width are set, the space required for the image is reserved when the page is loaded. Without these attributes, the browser does not know the size of the image, and cannot reserve the appropriate space to it. The effect will be that the page layout will change during loading (while the images load).

Example

Define an image as the submit button, with height and width attributes:

```
<form>
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="image" src="img_submit.gif" alt="Submit" width="48" height="48">
</form>
```

The list Attribute

The `input` `list` attribute refers to a `<datalist>` element that contains pre-defined options for an `<input>` element.

Example

An `<input>` element with pre-defined values in a `<datalist>`:

```
<form>
  <input list="browsers">
  <datalist id="browsers">
    <option value="Internet Explorer">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
</form>
```

The autocomplete Attribute

The `input autocomplete` attribute specifies whether a form or an input field should have autocomplete on or off.

Autocomplete allows the browser to predict the value. When a user starts to type in a field, the browser should display options to fill in the field, based on earlier typed values.

The `autocomplete` attribute works with `<form>` and the following `<input>` types: text, search, url, tel, email, password, datepickers, range, and color.

Example

An HTML form with autocomplete on, and off for one input field:

```
<form action="/action_page.php" autocomplete="on">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email" autocomplete="off"><br><br>
  <input type="submit" value="Submit">
</form>
```

Tip: In some browsers you may need to activate an autocomplete function for this to work (Look under "Preferences" in the browser's menu)

HTML Input form* Attributes

This chapter describes the different `form*` attributes for the HTML `<input>` element.

The form Attribute

The `input form` attribute specifies the form the `<input>` element belongs to.

The value of this attribute must be equal to the `id` attribute of the `<form>` element it belongs to.

Example

An input field located outside of the HTML form (but still a part of the form):

```
<form action="/action_page.php" id="form1">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <input type="submit" value="Submit">
</form>

<label for="lname">Last name:</label>
<input type="text" id="lname" name="lname" form="form1">
```

The formaction Attribute

The input `formaction` attribute specifies the URL of the file that will process the input when the form is submitted.

Note: This attribute overrides the `action` attribute of the `<form>` element.

The `formaction` attribute works with the following input types: submit and image.

Example

An HTML form with two submit buttons, with different actions:

```
<form action="/action_page.php">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="submit" value="Submit">
  <input type="submit" formaction="/action_page2.php" value="Submit as Admin">
</form>
```

The formenctype Attribute

The input `formenctype` attribute specifies how the form-data should be encoded when submitted (only for forms with `method="post"`).

Note: This attribute overrides the `enctype` attribute of the `<form>` element.

The `formenctype` attribute works with the following input types: submit and image.

Example

A form with two submit buttons. The first sends the form-data with default encoding, the second sends the form-data encoded as "multipart/form-data":

```
<form action="/action_page_binary.asp" method="post">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <input type="submit" value="Submit">
  <input type="submit" formenctype="multipart/form-data"
  value="Submit as Multipart/form-data">
</form>
```

ADVERTISEMENT

The formmethod Attribute

The input `formmethod` attribute defines the HTTP method for sending form-data to the action URL.

Note: This attribute overrides the method attribute of the `<form>` element.

The `formmethod` attribute works with the following input types: submit and image.

The form-data can be sent as URL variables (`method="get"`) or as an HTTP post transaction (`method="post"`).

Notes on the "get" method:

- This method appends the form-data to the URL in name/value pairs
- This method is useful for form submissions where a user want to bookmark the result
- There is a limit to how much data you can place in a URL (varies between browsers), therefore, you cannot be sure that all of the form-data will be correctly transferred
- Never use the "get" method to pass sensitive information! (password or other sensitive information will be visible in the browser's address bar)

Notes on the "post" method:

- This method sends the form-data as an HTTP post transaction
- Form submissions with the "post" method cannot be bookmarked
- The "post" method is more robust and secure than "get", and "post" does not have size limitations

Example

A form with two submit buttons. The first sends the form-data with `method="get"`. The second sends the form-data with `method="post"`:


```
<form action="/action_page.php" method="get">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="submit" value="Submit using GET">
  <input type="submit" formmethod="post" value="Submit using POST">
</form>
```

The formtarget Attribute

The input `formtarget` attribute specifies a name or a keyword that indicates where to display the response that is received after submitting the form.

Note: This attribute overrides the target attribute of the `<form>` element.

The `formtarget` attribute works with the following input types: submit and image.

Example

A form with two submit buttons, with different target windows:

```
<form action="/action_page.php">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="submit" value="Submit">
  <input type="submit" formtarget="_blank" value="Submit to a new window/tab">
</form>
```

The formnovalidate Attribute

The input `formnovalidate` attribute specifies that an `<input>` element should not be validated when submitted.

Note: This attribute overrides the novalidate attribute of the `<form>` element.

The `formnovalidate` attribute works with the following input types: submit.

Example

A form with two submit buttons (with and without validation):

```
<form action="/action_page.php">
  <label for="email">Enter your email:</label>
  <input type="email" id="email" name="email"><br><br>
```

```
<input type="submit" value="Submit">
<input type="submit" formnovalidate="formnovalidate"
value="Submit without validation">
</form>
```

The novalidate Attribute

The `novalidate` attribute is a `<form>` attribute.

When present, `novalidate` specifies that all of the form-data should not be validated when submitted.

Example

Specify that no form-data should be validated on submit:

```
<form action="/action_page.php" novalidate>
  <label for="email">Enter your email:</label>
  <input type="email" id="email" name="email"><br><br>
  <input type="submit" value="Submit">
</form>
```