**HTML:**

HTML is the standard markup language for Web pages

With HTML you can create your own Website.

Example

<!DOCTYPE html>  
<html>  
<head>  
<title>Page Title</title>  
</head>  
<body>  
  
<h1>This is a Heading</h1>  
<p>This is a paragraph.</p>  
  
</body>  
</html>

**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**

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

</head>  
<body>  
  
<h1>My First Heading</h1>

<p>Myfirstheading</h1>   
  
</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** |
| <h1> | My First Heading | </h1> |
| <p> | My first paragraph. | </p> |
| <br> | *none* | *none* |

## **Web Browsers**

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

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



## **HTML Page Structure**

Below is a visualization of an HTML page structure:

<html>

<head>

<title>Page title</title>

</head>

<body>

<h1>This is a heading</h1>

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

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

</body>

</html>

**Note:** The content inside the <body> section will be displayed in a browser. The content inside the <title> element will be shown in the browser's title bar or in the page's tab.

## **HTML History**

Since the early days of the World Wide Web, there have been many versions of HTML:

|  |  |
| --- | --- |
| **Year** | **Version** |
| 1989 | Tim Berners-Lee invented www |
| 1991 | Tim Berners-Lee invented HTML |
| 1993 | Dave Raggett drafted HTML+ |
| 1995 | HTML Working Group defined HTML 2.0 |
| 1997 | W3C Recommendation: HTML 3.2 |
| 1999 | W3C Recommendation: HTML 4.01 |
| 2000 | W3C Recommendation: XHTML 1.0 |
| 2008 | WHATWG HTML5 First Public Draft |
| 2012 | [WHATWG HTML5 Living Standard](http://whatwg.org/html/) |
| 2014 | [W3C Recommendation: HTML5](http://www.w3.org/TR/html5/) |
| 2016 | W3C Candidate Recommendation: HTML 5.1 |
| 2017 | [W3C Recommendation: HTML5.1 2nd Edition](http://www.w3.org/TR/html51/) |
| 2017 | [W3C Recommendation: HTML5.2](http://www.w3.org/TR/html52/) |

## **Web Browsers**

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

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



## **HTML Page Structure**

Below is a visualization of an HTML page structure:

<html>

<head>

<title>Page title</title>

</head>

<body>

<h1>This is a heading</h1>

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

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

</body>

</html>

# **HTML Editors**

## **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 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 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



**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>



Example

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

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

Output:

# This is a Heading

# This is a paragraph.

# **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>.

### Example

<!DOCTYPE html>  
<html>  
<body>  
  
<h1>My First Heading</h1>  
<p>My first paragraph.</p>  
  
</body>  
</html>

Output:

# My First Heading

My first paragraph.

## **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>

Output:

# This is heading 1

## This is heading 2

### This is heading 3

### This is heading 4

### This is heading 5

### This is heading 6

## **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>

<!DOCTYPE html>

<html>

<body>

<h2>HTML Links</h2>

<p>HTML links are defined with the a tag:</p>

<a href="https://www.w3schools.com">This is a link</a>

</body>

</html>

**Output:**

## HTML Links

HTML links are defined with the a tag:

[This is a link](https://www.w3schools.com/)

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 <img> tag.

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

### Example

<img src="w3schools.jpg" alt="W3Schools.com" width="104" height="142">

<!DOCTYPE html>

<html>

<body>

<h2>HTML Images</h2>

<p>HTML images are defined with the img tag:</p>

<img src="w3schools.jpg" alt="W3Schools.com" width="104" height="142">

</body>

</html>

Output:

## HTML Images

HTML images are defined with the img tag:



# **How to View HTML Source**

### View HTML Source Code:

Click CTRL + U in an HTML page, or right-click on the page and select "View Page Source". This will open a new tab containing the HTML source code of the page.

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

### Inspect an HTML Element:

Right-click on an element (or a blank area), and choose "Inspect" 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.

**HTML Elements:**

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

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

Examples of some HTML elements:

<h1>My First Heading</h1>

<p>My first paragraph.</p>

|  |  |  |
| --- | --- | --- |
| **Start tag** | **Element content** | **End tag** |
| <h1> | My First Heading | </h1> |
| <p> | My first paragraph. | </p> |
| <br> | none | none |

**Note:** Some HTML elements have no content (like the <br> 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>

Output:

# My First Heading

My first paragraph.

Example Explained

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

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

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

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

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

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

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

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

The <h1> element defines a heading.

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

<h1>My First Heading</h1>

The <p> element defines a paragraph.</p>

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>  
<p>This is a paragraph</p>  
  
</body>  
</html>

## **Empty HTML Elements**

HTML elements with no content are called empty elements.

The <br> 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**

W3Schools' tag reference contains additional information about these tags and their attributes.

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<html>](https://www.w3schools.com/tags/tag_html.asp) | Defines the root of an HTML document |
| [<body>](https://www.w3schools.com/tags/tag_body.asp) | Defines the document's body |
| [<h1> to <h6>](https://www.w3schools.com/tags/tag_hn.asp) | Defines HTML headings |

# **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

<!DOCTYPE html>

<html>

<body>

<h2>The href Attribute</h2><p>HTML links are defined with the a tag. The link address is specified in the href attribute:</p>

<a href="https://www.w3schools.com">Visit W3Schools</a>

</body>

</html>

## The href Attribute

HTML links are defined with the a tag. The link address is specified in the href attribute:

## **The src Attribute**

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

### Example

<img src="img\_girl.jpg">

<!DOCTYPE html>

<html>

<body>

<h2>The src Attribute</h2>

<p>HTML images are defined with the img tag, and the filename of the image source is specified in the src attribute:</p>

<img src="img\_girl.jpg" width="500" height="600">

</body>

</html>

Output:

## The src Attribute

HTML images are defined with the img tag, and the filename of the image source is specified in the src attribute:



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

**1. Absolute URL** - Links to an external image that is hosted on another website. Example: src="https://www.w3schools.com/images/img\_girl.jpg".

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

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

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

## **The width and height Attributes:**

The <img> tag should also contain the width and height attributes, which specify the width and height of the image (in pixels):

## **The alt Attribute:**

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

### Example

<img src="img\_girl.jpg" alt="Girl with a jacket">

## **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>

You can see all the language codes in our [HTML Language Code Reference](https://www.w3schools.com/tags/ref_language_codes.asp).

**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>

<!DOCTYPE html>

<html>

<body>

<h2 title="I'm a header">The title Attribute</h2>

<p title="I'm a tooltip">Mouse over this paragraph, to display the title attribute as a tooltip.</p>

</body>

</html>

**Output:**

## The title Attribute

Mouse over this paragraph, to display the title attribute as a tooltip.

## **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>

<!DOCTYPE html>

<html>

<body>

<h1>About W3Schools</h1>

<p title=About W3Schools>

You cannot omit quotes around an attribute value

if the value contains spaces.

</p>

<p><b>

If you move the mouse over the paragraph above,

your browser will only display the first word from the title.

</b></p>

</body>

</html>

**Output:**

You cannot omit quotes around an attribute value if the value contains spaces.

**If you move the mouse over the paragraph above, your browser will only display the first word from the title.**

## **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">

<!DOCTYPE html>

<html>

<body>

<h2>Single or Double Quotes?</h2>

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

<p>Move your mouse over the paragraphs below to see the effect:</p>

<p title='John "ShotGun" Nelson'>John with double quotes</p>

<p title="John 'ShotGun' Nelson">John with single quotes</p>

</body>

</html>

Output:

## Single or Double Quotes?

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

Move your mouse over the paragraphs below to see the effect:

John with double quotes

John with single quotes

## **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 <img> specifies the path to the image to be displayed
* The width and height attributes of <img> provide size information for images
* The alt attribute of <img> 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 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**

<!DOCTYPE html>

<html>

<body>

<h1>Heading 1</h1>

<h2>Heading 2</h2>

<h3>Heading 3</h3>

<h4>Heading 4</h4>

<h5>Heading 5</h5>

<h6>Heading 6</h6>

</body>

</html>

## **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>

# Heading 1

## Heading 2

### Heading 3

#### Heading 4

##### Heading 5

###### Heading 6

### 

<!DOCTYPE html>

<html>

<body>

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

<p>You can change the size of a heading with the style attribute, using the font-size property.</p>

</body>

</html>

## **HTML Tag Reference**

W3Schools' tag reference contains additional information about these tags and their attributes.

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<html>](https://www.w3schools.com/tags/tag_html.asp) | Defines the root of an HTML document |
| [<body>](https://www.w3schools.com/tags/tag_body.asp) | Defines the document's body |
| [<h1> to <h6>](https://www.w3schools.com/tags/tag_hn.asp) | Defines HTML headings |

## **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 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><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>

<!DOCTYPE html>

<html>

<body>

<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>

<p>

The number of lines in a paragraph depends on the size of the browser window. If you resize the browser window, the number of lines in this paragraph will change.

</p>

</body>

</html>

Output:

This paragraph contains a lot of lines in the source code, but the browser ignores it.

This paragraph contains a lot of spaces in the source code, but the browser ignores it.

The number of lines in a paragraph depends on the size of the browser window. If you resize the browser window, the number of lines in this paragraph will change.

## **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>

<!DOCTYPE html>

<html>

<body>

<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>

<h2>This is heading 2</h2>

<p>This is some other text.</p>

</body>

</html>

Output:

# This is heading 1

This is some text.

# **HTML Styles**

**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>

# <!DOCTYPE html>

# <html>

# <body>

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

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

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

# </body>

# </html>

Output:

This is a paragraph.

This is a paragraph.

This is a paragraph.

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

**HTML Line Breaks**

The HTML <br> element defines a line break.

Use <br> 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>

<!DOCTYPE html>

<html>

<body>

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

</body>

</html>

**Output:**

This is  
a paragraph  
with line breaks.

## 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>

<!DOCTYPE html>

<html>

<body>

<p>In HTML, spaces and new lines are ignored:</p>

<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>

</body>

</html>

**Output**:

In HTML, spaces and new lines are ignored:

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.

## **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 Tag Reference**

W3Schools' tag reference contains additional information about HTML elements and their attributes.

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<p>](https://www.w3schools.com/tags/tag_p.asp) | Defines a paragraph |
| [<hr>](https://www.w3schools.com/tags/tag_hr.asp) | Defines a thematic change in the content |
| [<br>](https://www.w3schools.com/tags/tag_br.asp) | Inserts a single line break |
| [<pre>](https://www.w3schools.com/tags/tag_pre.asp) | Defines pre-formatted text |

## **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>

<!DOCTYPE html>

<html>

<body style="background-color:powderblue;">

<h1>This is a heading</h1>

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

</body>

</html>

Output:

# This is a heading

This is a paragraph.

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>

<!DOCTYPE html>

<html>

<body>

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

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

</body>

</html>

Output:

# This is a heading

This is a paragraph.

# **HTML Text Formatting**

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

### Example

**This text is bold**

*This text is italic*

This issubscript and superscript

## **HTML Formatting Elements**

Formatting elements were designed to display special types of text:

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

**HTML <b> and <strong> Elements**

The HTML <b> element defines bold text, without any extra importance.

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

### Example

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

<!DOCTYPE html>

<html>

<body>

<p>This text is normal.</p>

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

</body>

</html>

Output:

This text is normal.

**This text is important!**

## **HTML <i> and <em> 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>

<!DOCTYPE html>

<html>

<body>

<p>This text is normal.</p>

<p><i>This text is italic.</i></p>

</body>

</html>

Output:This text is normal.

*This text is italic.*

## **HTML <small> Element**

The HTML <small> element defines smaller text:

### Example

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

**<!DOCTYPE html>**

**<html>**

**<body>**

**<p>This is some normal text.</p>**

**<p><small>This is some smaller text.</small></p>**

**</body>**

**</html>**

**Output:**

This is some normal text.

This is some smaller text.

## **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 <del> Element**

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

## **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>

### Example

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

<!DOCTYPE html>

<html>

<body>

<p>This is <sub>subscripted</sub> text.</p>

</body>

</html>

Output:

This is subscripted text.

# **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 60 years, WWF has worked to help people and nature thrive. As the world's leading conservation organization, WWF works in nearly 100 countries. At every level, we collaborate with people around the world to develop and deliver innovative solutions that protect communities, wildlife, and the places in which they live.

## **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 60 years, WWF has worked to help people and nature thrive. As the world's leading conservation organization, WWF works in nearly 100 countries. At every level, we collaborate with people around the world to develop and deliver innovative solutions that protect communities, wildlife, and the places in which they live.  
</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>

<!DOCTYPE html>

<html>

<body>

<p>Browsers usually insert quotation marks around the q element.</p>

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

</body>

</html>

Output:

Browsers usually insert quotation marks around the q element.

WWF's goal is to: Build a future where people live in harmony with nature.

## **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

### <!DOCTYPE html>

### <html>

### <body>

### <p>The HTML address element defines contact information (author/owner) of a document or article.</p>

### <address>

### Written by John Doe.<br>

### Visit us at:<br>

### Example.com<br>

### Box 564, Disneyland<br>

### USA

### </address>

### </body>

### </html>

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

*Written by John Doe.  
Visit us at:  
Example.com  
Box 564, Disneyland  
USA*

Output:

The HTML address element defines contact information (author/owner) of a document or article.

## **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 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 -->

<!DOCTYPE html>

<html>

<body>

<!-- This is a comment -->

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

<!-- Comments are not displayed in the browser -->

</body>

</html>

Output:

This is a paragraph.

## **Hide Content**

## Comments can be used to hide content.

This can be helpful if you hide content temporarily:

<p>This is a paragraph.</p>  
  
<!-- <p>This is another paragraph </p> -->  
  
<p>This is a paragraph too.</p>

<!DOCTYPE html>

<html>

<body>

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

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

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

</body>

</html>

Output:

This is a paragraph.

This is a paragraph too.

<p>This is a paragraph.</p>  
<!--  
<p>Look at this cool image:</p>  
<img border="0" src="pic\_trulli.jpg" alt="Trulli">  
-->  
<p>This is a paragraph too.</p>

<!DOCTYPE html>

<html>

<body>

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

<!--

<p>Look at this cool image:</p>

<img border="0" src="pic\_trulli.jpg" alt="Trulli">

-->

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

</body>

</html>

# **HTML Colors**

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

**Color Names**

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

Tomato

Orange

DodgerBlue

MediumSeaGreen

Gray

SlateBlue

Violet

LightGray

HTML supports [140 standard color names](https://www.w3schools.com/colors/colors_names.asp).

**Background Color:**

You can set the background color for HTML elements:

Hello World

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

### Example

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

**Text Color**

You can set the color of text:

**Hello World**

Lorem ipsum dolor sit amet, consectetuer 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:

## **Hello World**

## **Hello World**

## **Hello World**

### 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>

<!DOCTYPE html>

<html>

<body>

<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>

</body>

</html>

# Hello World

# Hello World

# Hello World

## **Color Values**

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

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

**rgb(255, 99, 71)**

**#ff6347**

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

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

**rg55, 99, 71, 0.5)**

**h**Example

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

<!DOCTYPE html>

<html>

<body>

<p>Same as color name "Tomato":</p>

<h1 style="background-color:rgb(255, 99, 71);">rgb(255, 99, 71)</h1>

<h1 style="background-color:#ff6347;">#ff6347</h1>

<h1 style="background-color:hsl(9, 100%, 64%);">hsl(9, 100%, 64%)</h1>

<p>Same as color name "Tomato", but 50% transparent:</p>

<h1 style="background-color:rgba(255, 99, 71, 0.5);">rgba(255, 99, 71, 0.5)</h1>

<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">hsla(9, 100%, 64%, 0.5)</h1>

<p>In addition to the predefined color names, colors can be specified using RGB, HEX, HSL, or even transparent colors using RGBA or HSLA color values.</p>

</body>

</html>

**Output:**

Same as color name "Tomato":

# rgb(255, 99, 71)

# #ff6347

# hsl(9, 100%, 64%)

Same as color name "Tomato", but 50% transparent:

# rgba(255, 99, 71, 0.5)

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

In addition to the predefined color names, colors can be specified using RGB, HEX, HSL, or even transparent colors using RGBA or HSLA color values.

# **HTML RGB and RGBA Colors:**

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

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

**RGB Color Values**

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

**rgb(red, green, blue)**

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

This means that there are 256 x 256 x 256 = 16777216 possible colors!

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

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

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

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

Experiment by mixing the RGB values below:

**rgb(255, 99, 71)**

RED

255

GREEN

99

BLUE

71

### Example

**rgb(255, 0, 0)**

**rgb(0, 0, 255)**

**rgb(60, 179, 113)**

**rgb(238, 130, 238)**

**rgb(255, 165, 0)**

**rgb(106, 90, 205)**

## **Shades of Gray**

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

### Example

**rgb(60, 60, 60)**

**rgb(100, 100, 100)**

**rgb(140, 140, 140)**

**rgb(180, 180, 180)**

**rgb(200, 200, 200)**

**rgb(240, 240, 240)**

[Try it Yourself »](https://www.w3schools.com/html/tryit.asp?filename=tryhtml_color_rgb_gray)

## **RGBA Color Values**

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

An RGBA color value is specified with:

**rgba(red, green, blue, alpha)**

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

Experiment by mixing the RGBA values below:

RED

255

GREEN

99

BLUE

71

ALPHA

0.5

### Example

### rgba(255, 99, 71, 0)

### rgba(255, 99, 71, 0.2)

## **HTML Exercises**

Top of Form

**Exercise:**

Insert the correct RGB color values to make the background color completely blue.

<p style="background-color:rgb(, , )">This is a paragraph.</p>

# **HTML HEX Colors**

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

**HEX Color Values**

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

**#rrggbb**

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

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

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

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

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

Experiment by mixing the HEX values below:

**#ff6347**

RED

ff

GREEN

63

BLUE

47

### Example

**#ff0000**

**#0000ff**

**#3cb371**

**#ee82ee**

**#ffa500**

**#6a5acd**

<!DOCTYPE html>

<html>

<body>

<h1 style="background-color:#ff0000;">#ff0000</h1>

<h1 style="background-color:#0000ff;">#0000ff</h1>

<h1 style="background-color:#3cb371;">#3cb371</h1>

<h1 style="background-color:#ee82ee;">#ee82ee</h1>

<h1 style="background-color:#ffa500;">#ffa500</h1>

<h1 style="background-color:#6a5acd;">#6a5acd</h1>

</body>

</html>

# #ff0000

# #0000ff

# #3cb371

# #ee82ee

# #ffa500

# #6a5acd

## **Shades of Gray: HTML Exercises**

Top of Form

## **Test Yourself With Exercises**

## **Exercise:**

Insert the correct HEX value to make the text color white.

<p style="color: #">This is a paragraph.</p>

Bottom of Form

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

### Example

**#404040**

**#686868**

**#a0a0a0**

**#bebebe**

**#dcdcdc**

**#f8f8f8**

# **HTML HSL and HSLA Colors**

HSL stands for hue, saturation, and lightness.

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

## **HSL Color Values**

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

**hsl(hue, saturation, lightness)**

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

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

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

Experiment by mixing the HSL values below:

**hsl(0, 100%, 50%)**

HUE

0

SATURATION

100%

LIGHTNESS

50%

### Example

**hsl(0, 100%, 50%)**

**hsl(240, 100%, 50%)**

**hsl(147, 50%, 47%)**

**hsl(300, 76%, 72%)**

**hsl(39, 100%, 50%)**

**hsl(248, 53%, 58%)**

### Saturation

Saturation can be described as the intensity of a color.

100% is pure color, no shades of gray.

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

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

### Example

**hsl(0, 100%, 50%)**

**hsl(0, 80%, 50%)**

**hsl(0, 60%, 50%)**

**hsl(0, 40%, 50%)**

**hsl(0, 20%, 50%)**

**hsl(0, 0%, 50%)**

### Lightness

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

### Example

**hsl(0, 100%, 0%)**

**hsl(0, 100%, 25%)**

**hsl(0, 100%, 50%)**

**hsl(0, 100%, 75%)**

**hsl(0, 100%, 90%)**

**hsl(0, 100%, 100%)**

## **Shades of Gray**

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

### Example

**hsl(0, 0%, 20%)**

**hsl(0, 0%, 30%)**

**hsl(0, 0%, 40%)**

**hsl(0, 0%, 60%)**

**hsl(0, 0%, 70%)**

**hsl(0, 0%, 90%)**

## **HSLA Color Values**

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

An HSLA color value is specified with:

**hsla(hue, saturation, lightness, alpha)**

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

Experiment by mixing the HSLA values below:

**hsla(0, 100%, 50%, 0.5)**

HUE

0

SATURATION

100%

LIGHTNESS

50%

ALPHA

0.5

### Example

### hsla(9, 100%, 64%, 0)

### hsla(9, 100%, 64%, 0.2) 100%, 64%, 0.4)

# **HTML Styles - CSS**

CSS stands for Cascading Style Sheets.

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

**CSS = Styles and Colors**

Manipulate Text

Colors,  Boxes

**What is CSS?**

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

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

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

## **Using CSS**

CSS can be added to HTML documents in 3 ways:

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

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

## **Inline CSS**

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

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

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

### Example

<h1 style="color:blue;">A Blue Heading</h1>  
  
<p style="color:red;">A red paragraph.</p> <!DOCTYPE html>

<html>

<body>

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

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

</body>

</html>

Output:

# A Blue Heading

A red paragraph.

## **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>

Output:

# This is a heading

This is a paragraph.

## **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>

Output:

# This is a heading

This is a paragraph.

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

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

### "styles.css":

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

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

## **CSS Colors, Fonts and Sizes**

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

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

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

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

### Example

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

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

Output:

# This is a heading

This is a paragraph.

## **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;  
}

<!DOCTYPE html>

<html>

<head>

<style>

p {

border: 2px solid powderblue;

}

</style>

</head>

<body>

<h1>This is a heading</h1>

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

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

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

</body>

</html>

Output:

# This is a heading

This is a paragraph.

This is a paragraph.

This is a paragraph.

## **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">

## **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">

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://www.w3schools.com/html/styles.css">

</head>

<body>

<h1>This is a heading</h1>

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

</body>

</html>

Output:

# This is a heading

This is a paragraph.

# **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>

<!DOCTYPE html>

<html>

<body>

<h1>HTML Links</h1>

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

</body>

</html>

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>

<!DOCTYPE html>

<html>

<body>

<h2>The target Attribute</h2>

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

<p>If target="\_blank", the link will open in a new browser window or tab.</p>

</body>

</html>

## **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>

<!DOCTYPE html>

<html>

<body>

<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>

</body>

</html>

Output:

## Absolute URLs

[W3C](https://www.w3.org/)

[Google](https://www.google.com/)

## Relative URLs

[HTML Images](https://www.w3schools.com/html/html_images.asp)

[CSS Tutorial](https://www.w3schools.com/css/default.asp)

## **HTML Links - Use an Image as a Link**

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

### Example

<a href="default.asp">  
<img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;">  
</a>

<!DOCTYPE html>

<html>

<body>

<h2>Image as a Link</h2>

<p>The image below is a link. Try to click on it.</p>

<a href="default.asp"><img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;"></a>

</body>

</html>

Output:

## Image as a Link

The image below is a link. Try to click on it.

## **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>

**<!DOCTYPE html>**

**<html>**

**<body>**

**<h2>Link to an Email Address</h2>**

**<p>To create a link that opens in the user's email program (to let them send a new email), use mailto: inside the href attribute:</p>**

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

**</body>**

**</html>**

**Output:**

## Link to an Email Address

To create a link that opens in the user's email program (to let them send a new email), use mailto: inside the href attribute:

[Send email](mailto:someone@example.com)

## **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:

<!DOCTYPE html>

<html>

<body>

<h2>Button as a Links</h2>

<p>Click the button to go to the HTML tutorial.</p>

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

</body>

</html>

Output:

## Button as a Links

Click the button to go to the HTML tutorial.

HTML Tutorial

## **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>

<!DOCTYPE html>

<html>

<body>

<h2>Button as a Links</h2>

<p>Click the button to go to the HTML tutorial.</p>

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

</body>

</html>

Output:

## Button as a Links

Click the button to go to the HTML tutorial.

HTML Tutorial

## **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 <img> 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 Link Tags**

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<a>](https://www.w3schools.com/tags/tag_a.asp) | Defines a hyperlink |

For a complete list of all available HTML tags, visit our [HTML Tag Reference](https://www.w3schools.com/tags/default.asp).

# **HTML Links - Different Colors**

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

**HTML Link Colors**

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

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

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

### Example

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

<style>  
a:link {  
  color: green;  
  background-color: transparent;  
  text-decoration: none;  
}  
  
a:visited {  
  color: pink;  
  background-color: transparent;  
  text-decoration: none;  
}  
  
a:hover {  
  color: red;  
  background-color: transparent;  
  text-decoration: underline;  
}  
  
a:active {  
  color: yellow;  
  background-color: transparent;  
  text-decoration: underline;  
}  
</style>

<!DOCTYPE html>

<html>

<head>

<style>

a:link {

color: green;

background-color: transparent;

text-decoration: none;

}

a:visited {

color: pink;

background-color: transparent;

text-decoration: none;

}

a:hover {

color: red;

background-color: transparent;

text-decoration: underline;

}

a:active {

color: yellow;

background-color: transparent;

text-decoration: underline;

}

</style>

</head>

<body>

<h2>Link Colors</h2>

<p>You can change the default colors of links</p>

<a href="html\_images.asp" target="\_blank">HTML Images</a>

</body>

</html>

Output:

## Link Colors

You can change the default colors of links

[HTML Images](https://www.w3schools.com/html/html_images.asp)

# **HTML Links - Create Bookmarks**

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

## **Create a Bookmark in HTML**

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

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

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

## **Example**

First, use the id attribute to create a bookmark:

<h2 id="C4">Chapter 4</h2>

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

### Example

<a href="#C4">Jump to Chapter 4</a>

<!DOCTYPE html>

<html>

<body>

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

<p><a href="#C10">Jump to Chapter 10</a></p>

<h2>Chapter 1</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 2</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 3</h2>

<p>This chapter explains ba bla bla</p>

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

<p>This chapter explains ba bla bla</p>

<h2>Chapter 5</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 6</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 7</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 8</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 9</h2>

<p>This chapter explains ba bla bla</p>

<h2 id="C10">Chapter 10</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 11</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 12</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 13</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 14</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 15</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 16</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 17</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 18</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 19</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 20</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 21</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 22</h2>

<p>This chapter explains ba bla bla</p>

<h2>Chapter 23</h2>

<p>This chapter explains ba bla bla</p>

</body>

</html>

Output:

[Jump to Chapter 4](https://www.w3schools.com/html/tryit.asp?filename=tryhtml_links_bookmark#C4)

[Jump to Chapter 10](https://www.w3schools.com/html/tryit.asp?filename=tryhtml_links_bookmark#C10)

## Chapter 1

This chapter explains ba bla bla

## Chapter 2

This chapter explains ba bla bla

## Chapter 3

This chapter explains ba bla bla

## Chapter 4

This chapter explains ba bla bla

## Chapter 5

This chapter explains ba bla bla

## Chapter 6

This chapter explains ba bla bla

## Chapter 7

This chapter explains ba bla bla

## Chapter 8

This chapter explains ba bla bla

## Chapter 9

This chapter explains ba bla bla

## Chapter 10

This chapter explains ba bla bla

## Chapter 11

This chapter explains ba bla bla

## Chapter 12

This chapter explains ba bla bla

## Chapter 13

This chapter explains ba bla bla

## Chapter 14

This chapter explains ba bla bla

## Chapter 15

This chapter explains ba bla bla

## Chapter 16

This chapter explains ba bla bla

## Chapter 17

This chapter explains ba bla bla

## Chapter 18

This chapter explains ba bla bla

## Chapter 19

This chapter explains ba bla bla

## Chapter 20

This chapter explains ba bla bla

## Chapter 21

This chapter explains ba bla bla

## Chapter 22

This chapter explains ba bla bla

## Chapter 23

This chapter explains ba bla bla

# **HTML Images**

<!DOCTYPE html>

<html>

<body>

<h2>HTML Image</h2>

<img src="pic\_trulli.jpg" alt="Trulli" width="500" height="333">

</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

<img src="/images/html5.gif" alt="HTML5 Icon" style="width:128px;height:128px;">

## 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

<img src="https://www.w3schools.com/images/w3schools\_green.jpg" alt="W3Schools.com">

**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; they can suddenly be removed or changed.

## Animated Images

HTML allows animated GIFs:

### Example

<img src="programming.gif" alt="Computer Man" style="width:48px;height:48px;">

## Image as a Link

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

### Example

<a href="default.asp">  
  <img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;">  
</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><img src="smiley.gif" alt="Smiley face" style="float:right;width:42px;height:42px;">  
The image will float to the right of the text.</p>  
  
<p><img src="smiley.gif" alt="Smiley face" style="float:left;width:42px;height:42px;">  
The image will float to the left of the text.</p>

**Tip:** To learn more about CSS Float, read our [CSS Float Tutorial](https://www.w3schools.com/css/css_float.asp).

## 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, .pjp |
| PNG | Portable Network Graphics | .png |
| SVG | Scalable Vector Graphics | .svg |

## Chapter Summary

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

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

## HTML Image Tags

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<img>](https://www.w3schools.com/tags/tag_img.asp) | Defines an image |
| [<map>](https://www.w3schools.com/tags/tag_map.asp) | Defines an image map |
| [<area>](https://www.w3schools.com/tags/tag_area.asp) | Defines a clickable area inside an image map |
| [<picture>](https://www.w3schools.com/tags/tag_picture.asp) | Defines a container for multiple image resources |

# **HTML Image Maps**

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

## Image Maps

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

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



### Example

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

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

## How Does it Work?

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

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

## The Image

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

<img src="workplace.jpg" alt="Workplace" usemap="#workmap">

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

**Tip:** You can use any image as an image map!

## Create Image Map

Then, add a <map> element.

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

<map name="workmap">

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

## The Areas

Then, add the clickable areas.

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

### Shape

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

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

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

### Shape="rect"

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

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



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



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

### Example

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

This is the area that becomes clickable and will send the user to the page "computer.htm":



### Shape="circle"

To add a circle area, first locate the coordinates of the center of the circle:

337,300



Then specify the radius of the circle:

44 pixels



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

### Example

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

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



### Shape="poly"

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

This can be used to create any shape.

Like maybe a croissant shape!

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



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



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

### Example

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

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



## Image Map and JavaScript

A clickable area can also trigger a JavaScript function.

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

### Example

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

<map name="workmap">  
  <area shape="circle" coords="337,300,44" href="coffee.htm" onclick="myFunction()">  
</map>  
  
<script>  
function myFunction() {  
  alert("You clicked the coffee cup!");  
}  
</script>

## Chapter Summary

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

Top of Form

## HTML Image Tags

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<img>](https://www.w3schools.com/tags/tag_img.asp) | Defines an image |
| [<map>](https://www.w3schools.com/tags/tag_map.asp) | Defines an image map |
| [<area>](https://www.w3schools.com/tags/tag_area.asp) | Defines a clickable area inside an image map |
| [<picture>](https://www.w3schools.com/tags/tag_picture.asp) | Defines a container for multiple image resources |

For a complete list of all available HTML tags, visit our [HTML Tag Reference](https://www.w3schools.com/tags/default.asp).

# **HTML Background Images**

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

## Background Image on a HTML element

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

### Example

Add a background image on a HTML element:

<p style="background-image: url('img\_girl.jpg');">

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

### Example

Specify the background image in the <style> element:

<style>  
p {  
  background-image: url('img\_girl.jpg');  
}  
</style>

## Background Image on a Page

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

### Example

Add a background image for the entire page:

<style>  
body {  
  background-image: url('img\_girl.jpg');  
}  
</style>

## Background Repeat

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

### Example

<style>  
body {  
  background-image: url('example\_img\_girl.jpg');  
}  
</style>

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

### Example

<style>  
body {  
  background-image: url('example\_img\_girl.jpg');  
  background-repeat: no-repeat;  
}  
</style>

## Background Cover

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

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

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

### Example

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

## Background Stretch

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

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

### Example

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

# **HTML <picture> Element**

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



## The HTML <picture> Element

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

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

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

### Example

Show different images for different screen sizes:

<picture>  
  <source media="(min-width: 650px)" srcset="img\_food.jpg">  
  <source media="(min-width: 465px)" srcset="img\_car.jpg">  
  <img src="img\_girl.jpg">  
</picture>

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

## When to use the Picture Element

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

### 1. Bandwidth

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

### 2. Format Support

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

### Example

The browser will use the first image format it recognizes:

<picture>  
  <source srcset="img\_avatar.png">  
  <source srcset="img\_girl.jpg">  
  <img src="img\_beatles.gif" alt="Beatles" style="width:auto;">  
</picture>

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

## HTML Image Tags

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<img>](https://www.w3schools.com/tags/tag_img.asp) | Defines an image |
| [<map>](https://www.w3schools.com/tags/tag_map.asp) | Defines an image map |
| [<area>](https://www.w3schools.com/tags/tag_area.asp) | Defines a clickable area inside an image map |
| [<picture>](https://www.w3schools.com/tags/tag_picture.asp) | Defines a container for multiple image resources |

# **HTML Favicon**

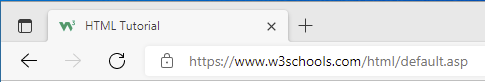
A favicon is a small image displayed next to the page title in the browser tab.

## 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](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>](https://www.w3schools.com/tags/tag_link.asp) | Defines the relationship between a document and an external resource |

# **HTML Page Title**

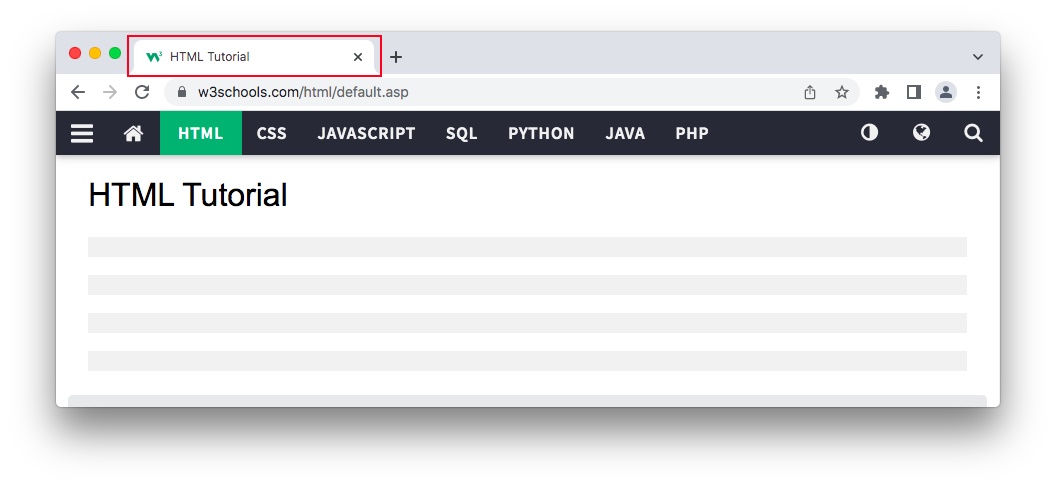
Every web page should have a page title to describe the meaning of the page.

The <title> element adds a title to your page:

### Example

<!DOCTYPE html>  
<html>  
<head>  
  <title>HTML Tutorial</title>  
</head>  
<body>  
  
The content of the document......  
  
</body>  
</html>

The title is shown in the browser's title bar:



The title should describe the content and the meaning of the page.

The page title is very important for search engine optimization (SEO). The text 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!

## HTML Title Tag

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<title>](https://www.w3schools.com/tags/tag_title.asp) | Defines the title of the document |

# **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:** A table cell can contain all sorts of HTML elements: text, images, lists, links, other tables, etc.

## Table Rows

Each table row starts with a <tr> and ends 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 when 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 table header cells. In those cases use the <th> tag instead of the <td> tag:

th stands for table header.

### Example

Let the first row be table header cells:

<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.

## HTML Table Tags

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<table>](https://www.w3schools.com/tags/tag_table.asp) | Defines a table |
| [<th>](https://www.w3schools.com/tags/tag_th.asp) | Defines a header cell in a table |
| [<tr>](https://www.w3schools.com/tags/tag_tr.asp) | Defines a row in a table |
| [<td>](https://www.w3schools.com/tags/tag_td.asp) | Defines a cell in a table |
| [<caption>](https://www.w3schools.com/tags/tag_caption.asp) | Defines a table caption |
| [<colgroup>](https://www.w3schools.com/tags/tag_colgroup.asp) | Specifies a group of one or more columns in a table for formatting |
| [<col>](https://www.w3schools.com/tags/tag_col.asp) | Specifies column properties for each column within a <colgroup> element |
| [<thead>](https://www.w3schools.com/tags/tag_thead.asp) | Groups the header content in a table |
| [<tbody>](https://www.w3schools.com/tags/tag_tbody.asp) | Groups the body content in a table |
| [<tfoot>](https://www.w3schools.com/tags/tag_tfoot.asp) | 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

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;  
}

## Round Table Borders

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

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

### Example

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

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

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

### Example

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

## 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;  
}

## 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 Table Headers**

HTML tables can have headers for each column or row, or for many columns/rows.

|  |  |  |
| --- | --- | --- |
| **EMIL** | **TOBIAS** | **LINUS** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **8:00** |  |  |
| **9:00** |  |  |
| **10:00** |  |  |
| **11:00** |  |  |
| **12:00** |  |  |
| **13:00** |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **MON** | **TUE** | **WED** | **THU** | **FRI** |
| **8:00** |  |  |  |  |  |
| **9:00** |  |  |  |  |  |
| **10:00** |  |  |  |  |  |
| **11:00** |  |  |  |  |  |
| **12:00** |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **DECEMBER** | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## HTML Table Headers

Table headers are defined with th elements. Each th element represents a table cell.

### Example

<table>  
  <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>

## Vertical Table Headers

To use the first column as table headers, define the first cell in each row as a <th> element:

### Example

<table>  
  <tr>  
    <th>Firstname</th>  
    <td>Jill</td>  
    <td>Eve</td>  
  </tr>  
  <tr>  
    <th>Lastname</th>  
    <td>Smith</td>  
    <td>Jackson</td>  
  </tr>  
  <tr>  
    <th>Age</th>  
    <td>94</td>  
    <td>50</td>  
  </tr>  
</table>

## Align Table Headers

By default, table headers are bold and centered:

|  |  |  |
| --- | --- | --- |
| **Firstname** | **Lastname** | **Age** |
| Jill | Smith | 50 |
| Eve | Jackson | 94 |

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

### Example

th {  
  text-align: left;  
}

## Header for Multiple Columns

You can have a header that spans over two or more columns.

|  |  |  |
| --- | --- | --- |
| **Name** | | **Age** |
| Jill | Smith | 50 |
| Eve | Jackson | 94 |

To do this, use the colspan attribute on the <th> element:

### Example

<table>  
  <tr>  
    <th colspan="2">Name</th>  
    <th>Age</th>  
  </tr>  
  <tr>  
    <td>Jill</td>  
    <td>Smith</td>  
    <td>50</td>  
  </tr>  
  <tr>  
    <td>Eve</td>  
    <td>Jackson</td>  
    <td>94</td>  
  </tr>  
</table>

You will learn more about colspan and rowspan in the [Table colspan & rowspan](https://www.w3schools.com/html/html_table_colspan_rowspan.asp) chapter.

## Table Caption

You can add a caption that serves as a heading for the entire table.

|  |  |
| --- | --- |
| Monthly savings | |
| **Month** | **Savings** |
| January | $100 |
| February | $50 |

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

### Example

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

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

# **HTML Table Padding & Spacing**

HTML tables can adjust the padding inside the cells, and also the space between the cells.

|  |  |  |
| --- | --- | --- |
| With Padding | | |
| hello | hello | hello |
| hello | hello | hello |
| hello | hello | hello |

|  |  |  |
| --- | --- | --- |
| With Spacing | | |
| hello | hello | hello |
| hello | hello | hello |
| hello | hello | hello |

## HTML Table - Cell Padding

Cell padding is the space between the cell edges and the cell content.

By default the padding is set to 0.

To add padding on table cells, use the CSS padding property:

### Example

th, td {  
  padding: 15px;  
}

To add padding only above the content, use the padding-top property.

And the others sides with the padding-bottom, padding-left, and padding-right properties:

### Example

th, td {  
  padding-top: 10px;  
  padding-bottom: 20px;  
  padding-left: 30px;  
  padding-right: 40px;  
}

## HTML Table - Cell Spacing

Cell spacing is the space between each cell.

By default the space is set to 2 pixels.

To change the space between table cells, use the CSS border-spacing property on the table element:

### Example

table {  
  border-spacing: 30px;  
}

# **HTML Table Colspan & Rowspan**

HTML tables can have cells that span over multiple rows and/or columns.

|  |  |  |
| --- | --- | --- |
| **NAME** | |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **APRIL** |  |  |
|  |  |
|  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **2022** | | |
|  |  |  |
| **FIESTA** | |  |
|  |
|  |  |  |

## HTML Table - Colspan

To make a cell span over multiple columns, use the colspan attribute:

### Example

<table>  
  <tr>  
    <th colspan="2">Name</th>  
    <th>Age</th>  
  </tr>  
  <tr>  
    <td>Jill</td>  
    <td>Smith</td>  
    <td>43</td>  
  </tr>  
  <tr>  
    <td>Eve</td>  
    <td>Jackson</td>  
    <td>57</td>  
  </tr>  
</table>

**Note:** The value of the colspan attribute represents the number of columns to span.

## HTML Table - Rowspan

To make a cell span over multiple rows, use the rowspan attribute:

### Example

<table>  
  <tr>  
    <th>Name</th>  
    <td>Jill</td>  
  </tr>  
  <tr>  
    <th rowspan="2">Phone</th>  
    <td>555-1234</td>  
  </tr>  
  <tr>  
    <td>555-8745</td>  
</tr>  
</table>

**Note:** The value of the rowspan attribute represents the number of rows to span.

# **HTML Table Styling**

Use CSS to make your tables look better.

## HTML Table - Zebra Stripes

If you add a background color on every other table row, you will get a nice zebra stripes effect.

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |
| 17 | 18 | 19 | 20 |

To style every other table row element, use the :nth-child(even) selector like this:

### Example

tr:nth-child(even) {  
  background-color: #D6EEEE;  
}

**Note:** If you use (odd) instead of (even), the styling will occur on row 1,3,5 etc. instead of 2,4,6 etc.

## HTML Table - Vertical Zebra Stripes

To make vertical zebra stripes, style every other column, instead of every other row.

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |
| 17 | 18 | 19 | 20 |

Set the :nth-child(even) for table data elements like this:

### Example

td:nth-child(even), th:nth-child(even) {  
  background-color: #D6EEEE;  
}

**Note:** Put the :nth-child() selector on both th and td elements if you want to have the styling on both headers and regular table cells.

## Combine Vertical and Horizontal Zebra Stripes

You can combine the styling from the two examples above and you will have stripes on every other row and every other column.

If you use a transparent color you will get an overlapping effect.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

Use an rgba() color to specify the transparency of the color:

### Example

tr:nth-child(even) {  
  background-color: rgba(150, 212, 212, 0.4);  
}  
  
th:nth-child(even),td:nth-child(even) {  
  background-color: rgba(150, 212, 212, 0.4);  
}

## Horizontal Dividers

| **First Name** | **Last Name** | **Savings** |
| --- | --- | --- |
| Peter | Griffin | $100 |
| Lois | Griffin | $150 |
| Joe | Swanson | $300 |

If you specify borders only at the bottom of each table row, you will have a table with horizontal dividers.

Add the border-bottom property to all tr elements to get horizontal dividers:

### Example

tr {  
  border-bottom: 1px solid #ddd;  
}

## Hoverable Table

Use the :hover selector on tr to highlight table rows on mouse over:

|  |  |  |
| --- | --- | --- |
| **First Name** | **Last Name** | **Savings** |
| Peter | Griffin | $100 |
| Lois | Griffin | $150 |
| Joe | Swanson | $300 |

### Example

tr:hover {background-color: #D6EEEE;}

# **HTML Table Colgroup**

The <colgroup> element is used to style specific columns of a table.

## HTML Table Colgroup

If you want to style the two first columns of a table, use the <colgroup> and <col> elements.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **MON** | **TUE** | **WED** | **THU** | **FRI** | **SAT** | **SUN** |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |

The <colgroup> element should be used as a container for the column specifications.

Each group is specified with a <col> element.

The span attribute specifies how many columns that get the style.

The style attribute specifies the style to give the columns.

**Note:** There is a very limited selection of [legal CSS properties for colgroups](https://www.w3schools.com/html/html_table_colgroup.asp#legalcss).

### Example

<table>  
  <colgroup>  
    <col span="2" style="background-color: #D6EEEE">  
  </colgroup>  
  <tr>  
    <th>MON</th>  
    <th>TUE</th>  
    <th>WED</th>  
    <th>THU</th>

**Note:** The <colgroup> tag must be a child of a <table> element and should be placed before any other table elements, like <thead>, <tr>, <td> etc., but after the <caption> element, if present.

## Legal CSS Properties

There is only a very limited selection of CSS properties that are allowed to be used in the colgroup:

[width](https://www.w3schools.com/cssref/pr_dim_width.php) property  
[visibility](https://www.w3schools.com/cssref/pr_class_visibility.php) property  
[background](https://www.w3schools.com/cssref/css3_pr_background.php) properties  
[border](https://www.w3schools.com/cssref/pr_border.php) properties

All other CSS properties will have no effect on your tables.

## Multiple Col Elements

If you want to style more columns with different styles, use more <col> elements inside the <colgroup>:

### Example

<table>  
  <colgroup>  
    <col span="2" style="background-color: #D6EEEE">  
    <col span="3" style="background-color: pink">  
  </colgroup>  
  <tr>  
    <th>MON</th>  
    <th>TUE</th>  
    <th>WED</th>  
    <th>THU</th>

## Empty Colgroups

If you want to style columns in the middle of a table, insert a "empty" <col> element (with no styles) for the columns before:

### Example

<table>  
  <colgroup>  
    <col span="3">  
    <col span="2" style="background-color: pink">  
  </colgroup>  
  <tr>  
    <th>MON</th>  
    <th>TUE</th>  
    <th>WED</th>  
    <th>THU</th>

## Hide Columns

You can hide columns with the visibility: collapse property:

### Example

<table>  
  <colgroup>  
    <col span="2">  
    <col span="3" style="visibility: collapse">  
  </colgroup>  
  <tr>  
    <th>MON</th>  
    <th>TUE</th>  
    <th>WED</th>  
    <th>THU</th>

# **HTML Lists**

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

### Example

An unordered HTML list:

* Item
* Item
* Item
* Item

An ordered HTML list:

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

## Unordered HTML List

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

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

### Example

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

## Ordered HTML List

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

The list items will be marked with numbers by default:

### Example

<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** |
| [<ul>](https://www.w3schools.com/tags/tag_ul.asp) | Defines an unordered list |
| [<ol>](https://www.w3schools.com/tags/tag_ol.asp) | Defines an ordered list |
| [<li>](https://www.w3schools.com/tags/tag_li.asp) | Defines a list item |
| [<dl>](https://www.w3schools.com/tags/tag_dl.asp) | Defines a description list |
| [<dt>](https://www.w3schools.com/tags/tag_dt.asp) | Defines a term in a description list |
| [<dd>](https://www.w3schools.com/tags/tag_dd.asp) | Describes the term in a description list |

# **HTML Unordered Lists**

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

## Unordered HTML List

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

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

### Example

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

## Horizontal List with CSS

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

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

### Example

<!DOCTYPE html>  
<html>  
<head>  
<style>  
ul {  
  list-style-type: none;  
  margin: 0;  
  padding: 0;  
  overflow: hidden;  
  background-color: #333333;  
}  
  
li {  
  float: left;  
}  
  
li a {  
  display: block;  
  color: white;  
  text-align: center;  
  padding: 16px;  
  text-decoration: none;  
}  
  
li a:hover {  
  background-color: #111111;  
}  
</style>  
</head>  
<body>  
  
<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](https://www.w3schools.com/css/default.asp).

## Chapter Summary

* Use the HTML <ul> element to define an unordered list
* Use the CSS list-style-type property to define the list item marker
* Use the HTML <li> element to define a list item
* Lists can be nested
* List items can contain other HTML elements
* Use the CSS property float:left to display a list horizontally

## HTML List Tags

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<ul>](https://www.w3schools.com/tags/tag_ul.asp) | Defines an unordered list |
| [<ol>](https://www.w3schools.com/tags/tag_ol.asp) | Defines an ordered list |
| [<li>](https://www.w3schools.com/tags/tag_li.asp) | Defines a list item |
| [<dl>](https://www.w3schools.com/tags/tag_dl.asp) | Defines a description list |
| [<dt>](https://www.w3schools.com/tags/tag_dt.asp) | Defines a term in a description list |
| [<dd>](https://www.w3schools.com/tags/tag_dd.asp) | Describes the term in a description list |

# **HTML Ordered Lists**

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

## Ordered HTML List

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

The list items will be marked with numbers by default:

### Example

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

## Ordered HTML List - The Type Attribute

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

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

### Numbers:

<ol type="1">  
  <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 (<li>) can contain a new list, and other HTML elements, like images and links, etc.

## Chapter Summary

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

# **HTML Block and Inline Elements**

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

The two most common display values are 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>](https://www.w3schools.com/tags/tag_address.asp)

[<article>](https://www.w3schools.com/tags/tag_article.asp)

[<aside>](https://www.w3schools.com/tags/tag_aside.asp)

[<blockquote>](https://www.w3schools.com/tags/tag_blockquote.asp)

[<canvas>](https://www.w3schools.com/tags/tag_canvas.asp)

[<dd>](https://www.w3schools.com/tags/tag_dd.asp)

[<div>](https://www.w3schools.com/tags/tag_div.asp)

[<dl>](https://www.w3schools.com/tags/tag_dl.asp)

[<dt>](https://www.w3schools.com/tags/tag_dt.asp)

[<fieldset>](https://www.w3schools.com/tags/tag_fieldset.asp)

[<figcaption>](https://www.w3schools.com/tags/tag_figcaption.asp)

[<figure>](https://www.w3schools.com/tags/tag_figure.asp)

[<footer>](https://www.w3schools.com/tags/tag_footer.asp)

[<form>](https://www.w3schools.com/tags/tag_form.asp)

[<h1>-<h6>](https://www.w3schools.com/tags/tag_hn.asp)

[<header>](https://www.w3schools.com/tags/tag_header.asp)

[<hr>](https://www.w3schools.com/tags/tag_hr.asp)

[<li>](https://www.w3schools.com/tags/tag_li.asp)

[<main>](https://www.w3schools.com/tags/tag_main.asp)

[<nav>](https://www.w3schools.com/tags/tag_nav.asp)

[<noscript>](https://www.w3schools.com/tags/tag_noscript.asp)

[<ol>](https://www.w3schools.com/tags/tag_ol.asp)

[<p>](https://www.w3schools.com/tags/tag_p.asp)

[<pre>](https://www.w3schools.com/tags/tag_pre.asp)

[<section>](https://www.w3schools.com/tags/tag_section.asp)

[<table>](https://www.w3schools.com/tags/tag_table.asp)

[<tfoot>](https://www.w3schools.com/tags/tag_tfoot.asp)

[<ul>](https://www.w3schools.com/tags/tag_ul.asp)

[<video>](https://www.w3schools.com/tags/tag_video.asp)

## 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 <span> element inside a paragraph.

### Example

<span>Hello World</span>

Here are the inline elements in HTML:

[<a>](https://www.w3schools.com/tags/tag_a.asp)

[<abbr>](https://www.w3schools.com/tags/tag_abbr.asp)

[<acronym>](https://www.w3schools.com/tags/tag_acronym.asp)

[<b>](https://www.w3schools.com/tags/tag_b.asp)

[<bdo>](https://www.w3schools.com/tags/tag_bdo.asp)

[<big>](https://www.w3schools.com/tags/tag_big.asp)

[<br>](https://www.w3schools.com/tags/tag_br.asp)

[<button>](https://www.w3schools.com/tags/tag_button.asp)

[<cite>](https://www.w3schools.com/tags/tag_cite.asp)

[<code>](https://www.w3schools.com/tags/tag_code.asp)

[<dfn>](https://www.w3schools.com/tags/tag_dfn.asp)

[<em>](https://www.w3schools.com/tags/tag_em.asp)

[<i>](https://www.w3schools.com/tags/tag_i.asp)

[<img>](https://www.w3schools.com/tags/tag_img.asp)

[<input>](https://www.w3schools.com/tags/tag_input.asp)

[<kbd>](https://www.w3schools.com/tags/tag_kbd.asp)

[<label>](https://www.w3schools.com/tags/tag_label.asp)

[<map>](https://www.w3schools.com/tags/tag_map.asp)

[<object>](https://www.w3schools.com/tags/tag_object.asp)

[<output>](https://www.w3schools.com/tags/tag_output.asp)

[<q>](https://www.w3schools.com/tags/tag_q.asp)

[<samp>](https://www.w3schools.com/tags/tag_samp.asp)

[<script>](https://www.w3schools.com/tags/tag_script.asp)

[<select>](https://www.w3schools.com/tags/tag_select.asp)

[<small>](https://www.w3schools.com/tags/tag_small.asp)

[<span>](https://www.w3schools.com/tags/tag_span.asp)

[<strong>](https://www.w3schools.com/tags/tag_strong.asp)

[<sub>](https://www.w3schools.com/tags/tag_sub.asp)

[<sup>](https://www.w3schools.com/tags/tag_sup.asp)

[<textarea>](https://www.w3schools.com/tags/tag_textarea.asp)

[<time>](https://www.w3schools.com/tags/tag_time.asp)

[<tt>](https://www.w3schools.com/tags/tag_tt.asp)

[<var>](https://www.w3schools.com/tags/tag_var.asp)

**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 <span> Element

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

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

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

### Example

<p>My mother has <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

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

## HTML Tags

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<div>](https://www.w3schools.com/tags/tag_div.asp) | Defines a section in a document (block-level) |
| [<span>](https://www.w3schools.com/tags/tag_span.asp) | Defines a section in a document (inline) |

# **HTML Div Element**

The <div> element is used as a container for other HTML elements.

## The <div> Element

The <div> element is by default a block element, meaning that it takes all available width, and comes with line breaks before and after.

### Example

A <div> element takes up all available width:

Lorem Ipsum <div>I am a div</div> dolor sit amet.

### Result

Lorem Ipsum

I am a div

dolor sit amet.

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

## <div> as a container

The <div> element is often used to group sections of a web page together.

### Example

A <div> element with HTML elements:

<div>  
  <h2>London</h2>  
  <p>London is the capital city of England.</p>  
  <p>London has over 13 million inhabitants.</p>  
</div>

### Result

## London

London is the capital city of England.

London has over 13 million inhabitants.

## Center align a <div> element

If you have a <div> element that is not 100% wide, and you want to center-align it, set the CSS margin property to auto.

### Example

<style>  
div {  
  width:300px;  
  margin:auto;  
}  
</style>

### Result

## London

London is the capital city of England.

London has over 13 million inhabitants.

## Multiple <div> elements

You can have many <div> containers on the same page.

### Example

<div>  
  <h2>London</h2>  
  <p>London is the capital city of England.</p>  
  <p>London has over 13 million inhabitants.</p>  
</div>  
  
<div>  
  <h2>Oslo</h2>  
  <p>Oslo is the capital city of Norway.</p>  
  <p>Oslo has over 600.000 inhabitants.</p>  
</div>  
  
<div>  
  <h2>Rome</h2>  
  <p>Rome is the capital city of Italy.</p>  
  <p>Rome has almost 3 million inhabitants.</p>  
</div>

### Result

## London

London is the capital city of England.

London has over 13 million inhabitants.

## Oslo

Oslo is the capital city of Norway.

Oslo has over 600.000 inhabitants.

## Rome

Rome is the capital city of Italy.

Rome has almost 3 million inhabitants.

## Aligning <div> elements side by side

When building web pages, you often want to have two or more <div> elements side by side, like this:

## London

London is the capital city of England.

London has over 13 million inhabitants.

## Oslo

Oslo is the capital city of Norway.

Oslo has over 600.000 inhabitants.

## Rome

Rome is the capital city of Italy.

Rome has almost 3 million inhabitants.

There are different methods for aligning elements side by side, all include some CSS styling. We will look at the most common methods:

## Float

The CSS float property was not originally meant to align <div> elements side-by-side, but has been used for this purpose for many years.

The CSS float property is used for positioning and formatting content and allow elements float next to each other instead of on top of each other.

### Example

How to use float to align div elements side by side:

<style>  
.mycontainer {  
  width:100%;  
  overflow:auto;  
}  
.mycontainer div {  
  width:33%;  
  float:left;  
}  
</style>

### Result

## London

London is the capital city of England.

London has over 13 million inhabitants.

## Oslo

Oslo is the capital city of Norway.

Oslo has over 600.000 inhabitants.

## Rome

Rome is the capital city of Italy.

Rome has almost 3 million inhabitants.

## Inline-block

If you change the <div> element's display property from block to inline-block, the <div> elements will no longer add a line break before and after, and will be displayed side by side instead of on top of each other.

### Example

How to use display: inline-block to align div elements side by side:

<style>  
div {  
  width: 30%;  
  display: inline-block;  
}  
</style>

### Result

## London

London is the capital city of England.

London has over 13 million inhabitants.

## Oslo

Oslo is the capital city of Norway.

Oslo has over 600.000 inhabitants.

## Rome

Rome is the capital city of Italy.

Rome has almost 3 million inhabitants.

## Flex

The CSS Flexbox Layout Module was introduced to make it easier to design flexible responsive layout structure without using float or positioning.

To make the CSS flex method work, surround the <div> elements with another <div> element and give it the status as a flex container.

### Example

How to use flex to align div elements side by side:

<style>  
.mycontainer {  
  display: flex;  
}  
.mycontainer > div {  
  width:33%;  
}  
</style>

### Result

## London

London is the capital city of England.

London has over 13 million inhabitants.

## Oslo

Oslo is the capital city of Norway.

Oslo has over 600.000 inhabitants.

## Rome

Rome is the capital city of Italy.

Rome has almost 3 million inhabitants.

Grid

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

Sounds almost the same as flex, but has the ability to define more than one row and position each row individually.

The CSS grid method requires that you surround the <div> elements with another <div> element and give the status as a grid container, and you must specify the width of each column.

### Example

How to use grid to align <div> elements side by side:

<style>  
.grid-container {  
  display: grid;  
  grid-template-columns: 33% 33% 33%;  
}  
</style>

### Result

## London

London is the capital city of England.

London has over 13 million inhabitants.

## Oslo

Oslo is the capital city of Norway.

Oslo has over 600.000 inhabitants.

## Rome

Rome is the capital city of Italy.

Rome has almost 3 million inhabitants.

## HTML Tags

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<div>](https://www.w3schools.com/tags/tag_div.asp) | Defines a section in a document (block-level) |

# **HTML class Attribute**

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

Multiple HTML elements can share the same class.

## Using The class Attribute

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

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

### Example

<!DOCTYPE html>  
<html>  
<head>  
<style>  
.city {  
  background-color: tomato;  
  color: white;  
  border: 2px solid black;  
  margin: 20px;  
  padding: 20px;  
}  
</style>  
</head>  
<body>  
  
<div class="city">  
  <h2>London</h2>  
  <p>London is the capital of England.</p>  
</div>  
  
<div class="city">  
  <h2>Paris</h2>  
  <p>Paris is the capital of France.</p>  
</div>  
  
<div class="city">  
  <h2>Tokyo</h2>  
  <p>Tokyo is the capital of Japan.</p>  
</div>  
  
</body>  
</html>

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

### Example

<!DOCTYPE html>  
<html>  
<head>  
<style>  
.note {  
  font-size: 120%;  
  color: red;  
}  
</style>  
</head>  
<body>  
  
<h1>My <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!

## 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> point to the "city" class and will share the same style:

### Example

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

## Use of The class Attribute in JavaScript

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

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

### Example

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

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

## Chapter Summary

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

# **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>

## 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>

## 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 specify the size of the iframe
* Use border:none; to remove the border around the iframe

## HTML iframe Tag

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp) | Defines an inline frame |

# **HTML JavaScript**

JavaScript makes HTML pages more dynamic and interactive.

### Example

## My First JavaScript

Click me to display Date and Time

## The HTML <script> Tag

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

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

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

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

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

### Example

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

## A Taste of JavaScript

Here are some examples of what JavaScript can do:

### Example

JavaScript can change content:

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

### Example

JavaScript can change styles:

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

### Example

JavaScript can change attributes:

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

## The HTML <noscript> Tag

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

### Example

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

p> element to "Hello World!".

## HTML Script Tags

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<script>](https://www.w3schools.com/tags/tag_script.asp) | Defines a client-side script |
| [<noscript>](https://www.w3schools.com/tags/tag_noscript.asp) | Defines an alternate content for users that do not support client-side scripts |

# **HTML File Paths**

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

## File Path Examples

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

## HTML File Paths

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

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

* Web pages
* Images
* Style sheets
* JavaScripts

## Absolute File Paths

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

### Example

<img src="https://www.w3schools.com/images/picture.jpg" alt="Mountain">

The <img> tag is explained in the chapter: [HTML Images](https://www.w3schools.com/html/html_images.asp).

## Relative File Paths

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

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

### Example

<img src="/images/picture.jpg" alt="Mountain">

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

### Example

<img src="images/picture.jpg" alt="Mountain">

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

### Example

<img src="../images/picture.jpg" alt="Mountain">

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

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

* Web pages
* Images
* Style sheets
* JavaScripts

## **Absolute File Paths**

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

### Example

<img src="https://www.w3schools.com/images/picture.jpg" alt="Mountain">

The <img> tag is explained in the chapter: [HTML Images](https://www.w3schools.com/html/html_images.asp).

## **Relative File Paths**

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

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

### Example

<img src="/images/picture.jpg" alt="Mountain">

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

### Example

<img src="images/picture.jpg" alt="Mountain">

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

### Example

<img src="../images/picture.jpg" alt="Mountain">

# **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 content 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](https://www.w3schools.com/css/default.asp).

## **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 is 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.

[[](https://www.w3schools.com/html/example_withoutviewport.htm)  
  
**Without the viewport meta tag**](https://www.w3schools.com/html/example_withoutviewport.htm)

[[](https://www.w3schools.com/html/example_withviewport.htm)  
  
**With the viewport meta tag**](https://www.w3schools.com/html/example_withviewport.htm)

## **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](https://www.w3schools.com/js/default.asp).

## **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>  
<img src="images/stickman.gif" width="24" height="39" alt="Stickman">  
<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** |
| [<head>](https://www.w3schools.com/tags/tag_head.asp) | Defines information about the document |
| [<title>](https://www.w3schools.com/tags/tag_title.asp) | Defines the title of a document |
| [<base>](https://www.w3schools.com/tags/tag_base.asp) | Defines a default address or a default target for all links on a page |
| [<link>](https://www.w3schools.com/tags/tag_link.asp) | Defines the relationship between a document and an external resource |
| [<meta>](https://www.w3schools.com/tags/tag_meta.asp) | Defines metadata about an HTML document |
| [<script>](https://www.w3schools.com/tags/tag_script.asp) | Defines a client-side script |
| [<style>](https://www.w3schools.com/tags/tag_style.asp) | 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](javascript:void(0))
* [Paris](javascript:void(0))
* [Tokyo](javascript:void(0))

# **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:

|  |  |
| --- | --- |
| HTML5 Semantic Elements | * <header> - Defines a header for a document or a section * <nav> - Defines a set of navigation links * <section> - Defines a section in a document * <article> - Defines an independent, self-contained content * <aside> - Defines content aside from the content (like a sidebar) * <footer> - Defines a footer for a document or a section * <details> - Defines additional details that the user can open and close on demand * <summary> - Defines a heading for the <details> element |

## **HTML Layout Techniques**

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

* CSS framework
* CSS float property
* CSS flexbox
* CSS grid

## **CSS Frameworks**

If you want to create your layout fast, you can use a CSS framework

## **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](https://www.w3schools.com/css/css_float.asp) chapter.

### Example

## Cities

* [London](javascript:void(0))
* [Paris](javascript:void(0))
* [Tokyo](javascript:void(0))

# **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](https://www.w3schools.com/css/css3_flexbox.asp) chapter.

### Example

## Cities

* [London](https://www.w3schools.com/html/html_layout.asp)
* [Paris](https://www.w3schools.com/html/html_layout.asp)
* [Tokyo](https://www.w3schools.com/html/html_layout.asp)

# **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.

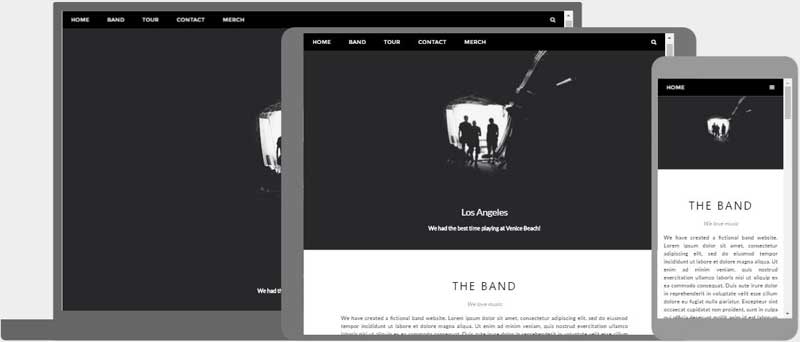
## **CSS Grid Layout**

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

# **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:  
[](https://www.w3schools.com/html/example_withoutviewport.htm)

With the viewport meta tag:  
[](https://www.w3schools.com/html/example_withviewport.htm)

## **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

<img src="img\_girl.jpg" **style="width:100%;"**>

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

<img src="img\_girl.jpg" style="**max-width:100%;**height:auto;">

### 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 changes 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">  
  <img src="img\_smallflower.jpg" alt="Flowers">  
</picture>

## **Responsive Text Size**

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

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

# **Hello World**

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

### Example

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

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

## **Media Queries**

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

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

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

Left Menu

Main Content

Right Content

### Example

<style>  
.left, .right {  
  float: left;  
  width: 20%; /\* The width is 20%, by default \*/  
}  
  
.main {  
  float: left;  
  width: 60%; /\* The width is 60%, by default \*/  
}  
  
/\* Use a media query to add a breakpoint at 800px: \*/  
@media screen and (max-width: 800px) {  
  .left, .main, .right {  
    width: 100%; /\* The width is 100%, when the viewport is 800px or smaller \*/  
  }  
}  
</style>

## **Responsive Web Design - Frameworks**

All popular CSS Frameworks offer responsive design.

They are free, and easy to use.

## **W3.CSS**

W3.CSS is a modern CSS framework with support for desktop, tablet, and mobile design by default.

W3.CSS is smaller and faster than similar CSS frameworks.

W3.CSS is designed to be independent of jQuery or any other JavaScript library.

# **W3.CSS Demo**

Resize the page to see the responsiveness!

## **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.

## **Paris**

Paris is the capital of France.

The Paris area is one of the largest population centers in Europe, with more than 12 million inhabitants.

## **Tokyo**

Tokyo is the capital of Japan.

It is the center of the Greater Tokyo Area, and the most populous metropolitan area in the world.

### Example

<!DOCTYPE html>  
<html>  
<meta name="viewport" content="width=device-width, initial-scale=1">  
<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">  
<body>  
  
<div class="w3-container w3-green">  
  <h1>W3Schools Demo</h1>  
  <p>Resize this responsive page!</p>  
</div>  
  
<div class="w3-row-padding">  
  <div class="w3-third">  
    <h2>London</h2>  
    <p>London is the capital city of England.</p>  
    <p>It is the most populous city in the United Kingdom,  
    with a metropolitan area of over 13 million inhabitants.</p>  
  </div>  
  
  <div class="w3-third">  
    <h2>Paris</h2>  
    <p>Paris is the capital of France.</p>  
    <p>The Paris area is one of the largest population centers in Europe,  
    with more than 12 million inhabitants.</p>  
  </div>  
  
  <div class="w3-third">  
    <h2>Tokyo</h2>  
    <p>Tokyo is the capital of Japan.</p>  
    <p>It is the center of the Greater Tokyo Area,  
    and the most populous metropolitan area in the world.</p>  
  </div>  
</div>  
  
</body>  
</html>

## **Bootstrap**

Another popular CSS framework is Bootstrap:

### Example

<!DOCTYPE html>  
<html lang="en">  
<head>  
<title>Bootstrap 5 Example</title>  
<meta charset="utf-8">  
<meta name="viewport" content="width=device-width, initial-scale=1">  
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet">  
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js"></script>  
</head>  
<body>  
  
<div class="container-fluid p-5 bg-primary text-white text-center">  
  <h1>My First Bootstrap Page</h1>  
  <p>Resize this responsive page to see the effect!</p>  
</div>  
  
<div class="container mt-5">  
  <div class="row">  
    <div class="col-sm-4">  
      <h3>Column 1</h3>  
      <p>Lorem ipsum...</p>  
    </div>  
    <div class="col-sm-4">  
      <h3>Column 2</h3>  
      <p>Lorem ipsum...</p>  
    </div>  
    <div class="col-sm-4">  
      <h3>Column 3</h3>  
      <p>Lorem ipsum...</p>  
    </div>  
  </div>  
</div>

# **HTML**Computer Code Elements

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

### Example

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

## **HTML <kbd> For Keyboard Input**

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

### Example

Define some text as keyboard input in a document:

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

Result:

Save the document by pressing Ctrl + S

## **HTML <samp> For Program Output**

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

### Example

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

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

Result:

Message from my computer:

File not found.  
Press F1 to continue

## **HTML <code> For Computer Code**

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

### Example

Define some text as computer code in a document:

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

Result:

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

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

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

### Example

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

Result:

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

## **HTML <var> For Variables**

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

### Example

Define some text as variables in a document:

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

Result:

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

## **Chapter Summary**

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

## **HTML Computer Code Elements**

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<code>](https://www.w3schools.com/tags/tag_code.asp) | Defines programming code |
| [<kbd>](https://www.w3schools.com/tags/tag_kbd.asp) | Defines keyboard input |
| [<samp>](https://www.w3schools.com/tags/tag_samp.asp) | Defines computer output |
| [<var>](https://www.w3schools.com/tags/tag_var.asp) | Defines a variable |
| [<pre>](https://www.w3schools.com/tags/tag_pre.asp) | Defines preformatted text |

# **HTML**Semantic Elements

Semantic elements = elements with a meaning.

## **What are Semantic Elements?**

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

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

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

## **Semantic Elements in HTML**

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

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

* <article>
* <aside>
* <details>
* <figcaption>
* <figure>
* <footer>
* <header>
* <main>
* <mark>
* <nav>
* <section>
* <summary>
* <time>



## **HTML <section> Element**

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

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

Examples of where a <section> element can be used:

* Chapters
* Introduction
* News items
* Contact information

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

### Example

Two sections in a document:

<section>  
<h1>WWF</h1>  
<p>The World Wide Fund for Nature (WWF) is an international organization working on issues regarding the conservation, research and restoration of the environment, formerly named the World Wildlife Fund. WWF was founded in 1961.</p>  
</section>  
  
<section>  
<h1>WWF's Panda symbol</h1>  
<p>The Panda has become the symbol of WWF. The well-known panda logo of WWF originated from a panda named Chi Chi that was transferred from the Beijing Zoo to the London Zoo in the same year of the establishment of WWF.</p>  
</section>

## **HTML <article> Element**

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

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

Examples of where the <article> element can be used:

* Forum posts
* Blog posts
* User comments
* Product cards
* Newspaper articles

### Example

Three articles with independent, self-contained content:

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

### Example 2

Use CSS to style the <article> element:

<html>  
<head>  
<style>  
.all-browsers {  
  margin: 0;  
  padding: 5px;  
  background-color: lightgray;  
}  
  
.all-browsers > h1, .browser {  
  margin: 10px;  
  padding: 5px;  
}  
  
.browser {  
  background: white;  
}  
  
.browser > h2, p {  
  margin: 4px;  
  font-size: 90%;  
}  
</style>  
</head>  
<body>  
  
<article class="all-browsers">  
  <h1>Most Popular Browsers</h1>  
  <article class="browser">  
    <h2>Google Chrome</h2>  
    <p>Google Chrome is a web browser developed by Google, released in 2008. Chrome is the world's most popular web browser today!</p>  
  </article>  
  <article class="browser">  
    <h2>Mozilla Firefox</h2>  
    <p>Mozilla Firefox is an open-source web browser developed by Mozilla. Firefox has been the second most popular web browser since January, 2018.</p>  
  </article>  
  <article class="browser">  
    <h2>Microsoft Edge</h2>  
    <p>Microsoft Edge is a web browser developed by Microsoft, released in 2015. Microsoft Edge replaced Internet Explorer.</p>  
  </article>  
</article>  
  
</body>  
</html>

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

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

The <section> element defines section in a document.

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

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

## **HTML <header> Element**

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

A <header> element typically contains:

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

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

### Example

A header for an <article>:

<article>  
  <header>  
    <h1>What Does WWF Do?</h1>  
    <p>WWF's mission:</p>  
  </header>  
  <p>WWF's mission is to stop the degradation of our planet's natural environment,  
  and build a future in which humans live in harmony with nature.</p>  
</article>

## **HTML <footer> Element**

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

A <footer> element typically contains:

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

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

### Example

A footer section in a document:

<footer>  
  <p>Author: Hege Refsnes</p>  
  <p><a href="mailto:hege@example.com">hege@example.com</a></p>  
</footer>

## **HTML <nav> Element**

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

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

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

### Example

A set of navigation links:

<nav>  
  <a href="/html/">HTML</a> |  
  <a href="/css/">CSS</a> |  
  <a href="/js/">JavaScript</a> |  
  <a href="/jquery/">jQuery</a>  
</nav>

## **HTML <aside> Element**

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

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

### Example

Display some content aside from the content it is placed in:

<p>My family and I visited The Epcot center this summer. The weather was nice, and Epcot was amazing! I had a great summer together with my family!</p>  
  
<aside>  
<h4>Epcot Center</h4>  
<p>Epcot is a theme park at Walt Disney World Resort featuring exciting attractions, international pavilions, award-winning fireworks and seasonal special events.</p>  
</aside>

### Example 2

Use CSS to style the <aside> element:

<html>  
<head>  
<style>  
aside {  
  width: 30%;  
  padding-left: 15px;  
  margin-left: 15px;  
  float: right;  
  font-style: italic;  
  background-color: lightgray;  
}  
</style>  
</head>  
<body>  
  
<p>My family and I visited The Epcot center this summer. The weather was nice, and Epcot was amazing! I had a great summer together with my family!</p>  
  
<aside>  
<p>The Epcot center is a theme park at Walt Disney World Resort featuring exciting attractions, international pavilions, award-winning fireworks and seasonal special events.</p>  
</aside>  
  
<p>My family and I visited The Epcot center this summer. The weather was nice, and Epcot was amazing! I had a great summer together with my family!</p>  
<p>My family and I visited The Epcot center this summer. The weather was nice, and Epcot was amazing! I had a great summer together with my family!</p>  
  
</body>  
</html>

## **HTML <figure> and <figcaption> Elements**

The <figure> tag specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.

The <figcaption> tag defines a caption for a <figure> element. The <figcaption> element can be placed as the first or as the last child of a <figure> element.

The <img> element defines the actual image/illustration.

### Example

<figure>  
  <img src="pic\_trulli.jpg" alt="Trulli">  
  <figcaption>Fig1. - Trulli, Puglia, Italy.</figcaption>  
</figure>

## **Why Semantic Elements?**

According to the W3C: "A semantic Web allows data to be shared and reused across applications, enterprises, and communities."

## **Semantic Elements in HTML**

Below is a list of some of the semantic elements in HTML.

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<article>](https://www.w3schools.com/tags/tag_article.asp) | Defines independent, self-contained content |
| [<aside>](https://www.w3schools.com/tags/tag_aside.asp) | Defines content aside from the page content |
| [<details>](https://www.w3schools.com/tags/tag_details.asp) | Defines additional details that the user can view or hide |
| [<figcaption>](https://www.w3schools.com/tags/tag_figcaption.asp) | Defines a caption for a <figure> element |
| [<figure>](https://www.w3schools.com/tags/tag_figure.asp) | Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc. |
| [<footer>](https://www.w3schools.com/tags/tag_footer.asp) | Defines a footer for a document or section |
| [<header>](https://www.w3schools.com/tags/tag_header.asp) | Specifies a header for a document or section |
| [<main>](https://www.w3schools.com/tags/tag_main.asp) | Specifies the main content of a document |
| [<mark>](https://www.w3schools.com/tags/tag_mark.asp) | Defines marked/highlighted text |
| [<nav>](https://www.w3schools.com/tags/tag_nav.asp) | Defines navigation links |
| [<section>](https://www.w3schools.com/tags/tag_section.asp) | Defines a section in a document |
| [<summary>](https://www.w3schools.com/tags/tag_summary.asp) | Defines a visible heading for a <details> element |
| [<time>](https://www.w3schools.com/tags/tag_time.asp) | Defines a date/time |

# **HTML**Style Guide

A consistent, clean, and tidy HTML code makes it easier for others to read and understand your code.

Here are some guidelines and tips for creating good HTML code.

## **Always Declare Document Type**

Always declare the document type as the first line in your document.

The correct document type for HTML is:

<!DOCTYPE html>

## **Use Lowercase Element Names**

HTML allows mixing uppercase and lowercase letters in element names.

However, we recommend using lowercase element names, because:

* Mixing uppercase and lowercase names looks bad
* Developers normally use lowercase names
* Lowercase looks cleaner
* Lowercase is easier to write

### Good:

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

### Bad:

<BODY>  
<P>This is a paragraph.</P>  
</BODY>

## **Close All HTML Elements**

In HTML, you do not have to close all elements (for example the <p> element).

However, we strongly recommend closing all HTML elements, like this:

### Good:

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

### Bad:

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

## **Use Lowercase Attribute Names**

HTML allows mixing uppercase and lowercase letters in attribute names.

However, we recommend using lowercase attribute names, because:

* Mixing uppercase and lowercase names looks bad
* Developers normally use lowercase names
* Lowercase looks cleaner
* Lowercase is easier to write

### 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>

## **Always Quote Attribute Values**

HTML allows attribute values without quotes.

However, we recommend quoting attribute values, because:

* Developers normally quote attribute values
* Quoted values are easier to read
* You MUST use quotes if the value contains spaces

### Good:

<table class="striped">

### Bad:

<table class=striped>

### Very bad:

This will not work, because the value contains spaces:

<table class=table striped>

## **Always Specify alt, width, and height for Images**

Always specify the alt attribute for images. This attribute is important if the image for some reason cannot be displayed.

Also, always define the width and height of images. This reduces flickering, because the browser can reserve space for the image before loading.

### Good:

<img src="html5.gif" alt="HTML5" style="width:128px;height:128px">

### Bad:

<img src="html5.gif">

## **Spaces and Equal Signs**

HTML allows spaces around equal signs. But space-less is easier to read and groups entities better together.

### Good:

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

### Bad:

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

## **Avoid Long Code Lines**

When using an HTML editor, it is NOT convenient to scroll right and left to read the HTML code.

Try to avoid too long code lines.

## **Blank Lines and Indentation**

Do not add blank lines, spaces, or indentations without a reason.

For readability, add blank lines to separate large or logical code blocks.

For readability, add two spaces of indentation. Do not use the tab key.

### Good:

<body>  
  
<h1>Famous Cities</h1>  
  
<h2>Tokyo</h2>  
<p>Tokyo is the capital of Japan, the center of the Greater Tokyo Area, and the most populous metropolitan area in the world.</p>  
  
<h2>London</h2>  
<p>London is the capital city of England. It is the most populous city in the United Kingdom.</p>  
  
<h2>Paris</h2>  
<p>Paris is the capital of France. The Paris area is one of the largest population centers in Europe.</p>  
  
</body>

### Bad:

<body>  
<h1>Famous Cities</h1>  
<h2>Tokyo</h2><p>Tokyo is the capital of Japan, the center of the Greater Tokyo Area, and the most populous metropolitan area in the world.</p>  
<h2>London</h2><p>London is the capital city of England. It is the most populous city in the United Kingdom.</p>  
<h2>Paris</h2><p>Paris is the capital of France. The Paris area is one of the largest population centers in Europe.</p>  
</body>

### Good Table Example:

<table>  
  <tr>  
    <th>Name</th>  
    <th>Description</th>  
  </tr>  
  <tr>  
    <td>A</td>  
    <td>Description of A</td>  
  </tr>  
  <tr>  
    <td>B</td>  
    <td>Description of B</td>  
  </tr>  
</table>

### Good List Example:

<ul>  
  <li>London</li>  
  <li>Paris</li>  
  <li>Tokyo</li>  
</ul>

## **Never Skip the <title> Element**

The <title> element is required in HTML.

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

The <title> element:

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

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

<title>HTML Style Guide and Coding Conventions</title>

## **Omitting <html> and <body>?**

An HTML page will validate without the <html> and <body> tags:

### Example

<!DOCTYPE html>  
<head>  
  <title>Page Title</title>  
</head>  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>

However, we strongly recommend to always add the <html> and <body> tags!

Omitting <body> can produce errors in older browsers.

Omitting <html> and <body> can also crash DOM and XML software.

## **Omitting <head>?**

The HTML <head> tag can also be omitted.

Browsers will add all elements before <body>, to a default <head> element.

### Example

<!DOCTYPE html>  
<html>  
<title>Page Title</title>  
<body>  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
  
</body>  
</html>

However, we recommend using the <head> tag.

## **Close Empty HTML Elements?**

In HTML, it is optional to close empty elements.

### Allowed:

<meta charset="utf-8">

### Also Allowed:

<meta charset="utf-8" />

If you expect XML/XHTML software to access your page, keep the closing slash (/), because it is required in XML and XHTML.

## **Add the lang Attribute**

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

### Example

<!DOCTYPE html>  
<html lang="en-us">  
<head>  
  <title>Page Title</title>  
</head>  
<body>  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
  
</body>  
</html>

## **Meta Data**

To ensure proper interpretation and correct search engine indexing, both the language and the character encoding <meta charset="charset"> should be defined as early as possible in an HTML document:

<!DOCTYPE html>  
<html lang="en-us">  
<head>  
  <meta charset="UTF-8">  
  <title>Page Title</title>  
</head>

## **Setting The Viewport**

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

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

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

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

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

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

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

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

[[](https://www.w3schools.com/html/example_withoutviewport.htm)  
  
**Without the viewport meta tag**](https://www.w3schools.com/html/example_withoutviewport.htm)

[[](https://www.w3schools.com/html/example_withviewport.htm)  
  
**With the viewport meta tag**](https://www.w3schools.com/html/example_withviewport.htm)

## **HTML Comments**

Short comments should be written on one line, like this:

<!-- This is a comment -->

Comments that spans more than one line, should be written like this:

<!--  
  This is a long comment example. This is a long comment example.  
  This is a long comment example. This is a long comment example.  
-->

Long comments are easier to observe if they are indented with two spaces.

## **Using Style Sheets**

Use simple syntax for linking to style sheets (the type attribute is not necessary):

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

Short CSS rules can be written compressed, like this:

p.intro {font-family:Verdana;font-size:16em;}

Long CSS rules should be written over multiple lines:

body {  
  background-color: lightgrey;  
  font-family: "Arial Black", Helvetica, sans-serif;  
  font-size: 16em;  
  color: black;  
}

* Place the opening bracket on the same line as the selector
* Use one space before the opening bracket
* Use two spaces of indentation
* Use semicolon after each property-value pair, including the last
* Only use quotes around values if the value contains spaces
* Place the closing bracket on a new line, without leading spaces

## **Loading JavaScript in HTML**

Use simple syntax for loading external scripts (the type attribute is not necessary):

<script src="myscript.js">

## **Accessing HTML Elements with JavaScript**

Using "untidy" HTML code can result in JavaScript errors.

These two JavaScript statements will produce different results:

### Example

getElementById("Demo").innerHTML = "Hello";  
  
getElementById("demo").innerHTML = "Hello";

## **Use Lower Case File Names**

Some web servers (Apache, Unix) are case sensitive about file names: "london.jpg" cannot be accessed as "London.jpg".

Other web servers (Microsoft, IIS) are not case sensitive: "london.jpg" can be accessed as "London.jpg".

If you use a mix of uppercase and lowercase, you have to be aware of this.

If you move from a case-insensitive to a case-sensitive server, even small errors will break your web!

To avoid these problems, always use lowercase file names!

## **File Extensions**

HTML files should have a **.html** extension (**.htm** is allowed).

CSS files should have a **.css** extension.

JavaScript files should have a **.js** extension.

## **Differences Between .htm and .html?**

There is no difference between the .htm and .html file extensions!

Both will be treated as HTML by any web browser and web server.

## **Default Filenames**

When a URL does not specify a filename at the end (like "https://www.w3schools.com/"), the server just adds a default filename, such as "index.html", "index.htm", "default.html", or "default.htm".

If your server is configured only with "index.html" as the default filename, your file must be named "index.html", and not "default.html".

However, servers can be configured with more than one default filename; usually you can set up as many default filenames as you want.

# **HTML**Entities

Reserved characters in HTML must be replaced with entities:

* < (less than) = **&lt;**
* > (greather than) = **&gt;**

## **HTML Character Entities**

Some characters are reserved in HTML.

If you use the less than (<) or greater than (>) signs in your HTML text, the browser might mix them with tags.

Entity names or entity numbers can be used to display reserved HTML characters.

Entity names look like this:

&*entity\_name*;

Entity numbers look like this:

&#*entity\_number*;

To display a less than sign (<) we must write: **&lt;** or **&#60;**

**Entity names** are easier to remember than entity numbers.

## **Non-breaking Space**

A commonly used HTML entity is the non-breaking space: **&nbsp;**

A non-breaking space is a space that will not break into a new line.

Two words separated by a non-breaking space will stick together (not break into a new line). This is handy when breaking the words might be disruptive.

Examples:

* § 10
* 10 km/h
* 10 PM

Another common use of the non-breaking space is to prevent browsers from truncating spaces in HTML pages.

If you write 10 spaces in your text, the browser will remove 9 of them. To add real spaces to your text, you can use the **&nbsp;** character entity.

The non-breaking hyphen ([&#8209;](https://www.w3schools.com/charsets/ref_utf_punctuation.asp)) is used to define a hyphen character (‑) that does not break into a new line.

## **Some Useful HTML Character Entities**

|  |  |  |  |
| --- | --- | --- | --- |
| **Result** | **Description** | **Name** | **Number** |
|  | non-breaking space | &nbsp; | &#160; |
| < | less than | &lt; | &#60; |
| > | greater than | &gt; | &#62; |
| & | ampersand | &amp; | &#38; |
| " | double quotation mark | &quot; | &#34; |
| ' | single quotation mark | &apos; | &#39; |
| ¢ | cent | &cent; | &#162; |
| £ | pound | &pound; | &#163; |
| ¥ | yen | &yen; | &#165; |
| € | euro | &euro; | &#8364; |
| © | copyright | &copy; | &#169; |
| ® | trademark | &reg; | &#174; |

## **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 be used in combination with alphanumeric characters to produce a character that is not present in the character set (encoding) used in the page.

Here are some examples:

|  |  |  |  |
| --- | --- | --- | --- |
| **Mark** | **Character** | **Construct** | **Result** |
| ̀ | a | a&#768; | à |
| ́ | a | a&#769; | á |
| ̂ | a | a&#770; | â |
| ̃ | a | a&#771; | ã |
| ̀ | O | O&#768; | Ò |
| ́ | O | O&#769; | Ó |
| ̂ | O | O&#770; | Ô |
| ̃ | O | O&#771; | Õ |

# **HTML**Symbols

Symbols or letters that are not present on your keyboard can be added to HTML using entities.

## **HTML Symbol Entities**

HTML entities were described in the previous chapter.

Many mathematical, technical, and currency symbols, are not present on a normal keyboard.

To add such symbols to an HTML page, you can use the entity name or the entity number (a decimal or a hexadecimal reference) for the symbol:

### Example

Display the euro sign:

<p>I will display &euro;</p>  
<p>I will display &#8364;</p>  
<p>I will display &#x20AC;</p>

### Will display as:

I will display €  
I will display €  
I will display €

## **Some Mathematical Symbols Supported by HTML**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Char** | **Number** | **Entity** | **Description** |  |
| ∀ | &#8704; | &forall; | For all |  |
| ∂ | &#8706; | &part; | Partial differential |  |
| ∃ | &#8707; | &exist; | There exists |  |
| ∅ | &#8709; | &empty; | Empty sets |  |
| ∇ | &#8711; | &nabla; | Nabla |  |
| ∈ | &#8712; | &isin; | Element of |  |
| ∉ | &#8713; | &notin; | Not an element of |  |
| ∋ | &#8715; | &ni; | Contains as member |  |
| ∏ | &#8719; | &prod; | N-ary product |  |
| ∑ | &#8721; | &sum; | N-ary summation |  |

## **Some Greek Letters Supported by HTML**

|  |  |  |  |
| --- | --- | --- | --- |
| **Char** | **Number** | **Entity** | **Description** |
| Α | &#913; | &Alpha; | GREEK ALPHA |
| Β | &#914; | &Beta; | GREEK BETA |
| Γ | &#915; | &Gamma; | GREEK GAMMA |
| Δ | &#916; | &Delta; | GREEK DELTA |
| Ε | &#917; | &Epsilon; | GREEK EPSILON |
| Ζ | &#918; | &Zeta; | GREEK ZETA |

## **Some Other Entities Supported by HTML**

|  |  |  |  |
| --- | --- | --- | --- |
| **Char** | **Number** | **Entity** | **Description** |
| © | &#169; | &copy; | COPYRIGHT |
| ® | &#174; | &reg; | REGISTERED |
| € | &#8364; | &euro; | EURO SIGN |
| ™ | &#8482; | &trade; | TRADEMARK |
| ← | &#8592; | &larr; | LEFT ARROW |
| ↑ | &#8593; | &uarr; | UP ARROW |
| → | &#8594; | &rarr; | RIGHT ARROW |
| ↓ | &#8595; | &darr; | DOWN ARROW |
| ♠ | &#9824; | &spades; | SPADE |
| ♣ | &#9827; | &clubs; | CLUB |
| ♥ | &#9829; | &hearts; | HEART |
| ♦ | &#9830; | &diams; | DIAMOND |

# **Using Emojis in HTML**

Emojis are characters from the UTF-8 character set: 😄 😍 💗

|  |  |
| --- | --- |
| **Emoji** | **Value** |
| 🗻 | &#128507; |
| 🗼 | &#128508; |
| 🗽 | &#128509; |
| 🗾 | &#128510; |
| 🗿 | &#128511; |
| 😀 | &#128512; |
| 😁 | &#128513; |
| 😂 | &#128514; |
| 😃 | &#128515; |
| 😄 | &#128516; |
| 😅 | &#128517; |

## **HTML Emojis Examples**

🚀🚁🚂🚃🚄

[HTML Emoji Transport Symbols](https://www.w3schools.com/charsets/ref_emoji_transport.asp)

💺💻💼💽💾

[HTML Emoji Office Symbols](https://www.w3schools.com/charsets/ref_emoji_office.asp)

👮👯👰👱👲

[HTML Emoji People Symbols](https://www.w3schools.com/charsets/ref_emoji_body.asp)

🐂🐃🐄🐅🐆

[HTML Emoji Animals Symbols](https://www.w3schools.com/charsets/ref_emoji_animals.asp)

## **What are Emojis?**

Emojis look like images, or icons, but they are not.

They are letters (characters) from the UTF-8 (Unicode) character set.

UTF-8 covers almost all of the characters and symbols in the world.

## **The HTML charset Attribute**

To display an HTML page correctly, a web browser must know the character set used in the page.

This is specified in the <meta> tag:

<meta charset="UTF-8">

If not specified, UTF-8 is the default character set in HTML.

## **UTF-8 Characters**

Many UTF-8 characters cannot be typed on a keyboard, but they can always be displayed using numbers (called entity numbers):

* A is 65
* B is 66
* C is 67

### Example

<!DOCTYPE html>  
<html>  
<meta charset="UTF-8">  
<body>  
  
<p>I will display A B C</p>  
<p>I will display &#65; &#66; &#67;</p>  
  
</body>  
</html>

### Example Explained

The <meta charset="UTF-8"> element defines the character set.

The characters A, B, and C, are displayed by the numbers 65, 66, and 67.

To let the browser understand that you are displaying a character, you must start the entity number with &# and end it with ; (semicolon).

## **Emoji Characters**

Emojis are also characters from the UTF-8 alphabet:

* 😄 is 128516
* 😍 is 128525
* 💗 is 128151

### Example

<!DOCTYPE html>  
<html>  
<meta charset="UTF-8">  
<body>  
  
<h1>My First Emoji</h1>  
  
<p>&#128512;</p>  
  
</body>  
</html>

Since Emojis are characters, they can be copied, displayed, and sized just like any other character in HTML.

### Example

<!DOCTYPE html>  
<html>  
<meta charset="UTF-8">  
<body>  
  
<h1>Sized Emojis</h1>  
  
<p style="font-size:48px">  
&#128512; &#128516; &#128525; &#128151;  
</p>  
  
</body>  
</html>

# **HTML**Encoding (Character Sets)

To display an HTML page correctly, a web browser must know which character set to use.

## **The HTML charset Attribute**

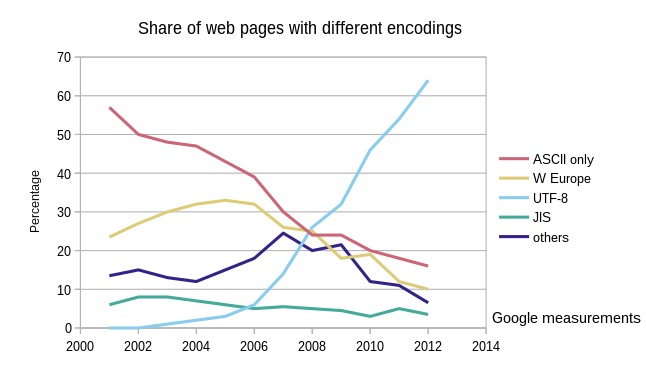
The character set is specified in the <meta> tag:

### Example

<meta charset="UTF-8">

The HTML5 specification encourages web developers to use the UTF-8 character set.

UTF-8 covers almost all of the characters and symbols in the world!

[](https://commons.wikimedia.org/wiki/File:Unicode_Web_growth.svg)

[Full UTF-8 Reference](https://www.w3schools.com/charsets/ref_utf_basic_latin.asp)

## **The ASCII Character Set**

ASCII was the first character encoding standard for the web. It defined 128 different characters that could be used on the internet:

* English letters (A-Z)
* Numbers (0-9)
* Special characters like ! $ + - ( ) @ < >.

## **The ANSI Character Set**

ANSI (Windows-1252) was the original Windows character set:

* Identical to ASCII for the first 127 characters
* Special characters from 128 to 159
* Identical to UTF-8 from 160 to 255

<meta charset="Windows-1252">

## **The ISO-8859-1 Character Set**

ISO-8859-1 was the default character set for HTML 4. This character set supported 256 different character codes. HTML 4 also supported UTF-8.

* Identical to ASCII for the first 127 characters
* Does not use the characters from 128 to 159
* Identical to ANSI and UTF-8 from 160 to 255

### HTML 4 Example

<meta http-equiv="Content-Type" content="text/html;charset=ISO-8859-1">

### HTML 5 Example

<meta charset="ISO-8859-1">

## **The UTF-8 Character Set**

* is identical to ASCII for the values from 0 to 127
* Does not use the characters from 128 to 159
* Identical to ANSI and 8859-1 from 160 to 255
* Continues from the value 256 to 10 000 characters

<meta charset="UTF-8">

## **Differences Between Character Sets**

The following table displays the differences between the character sets described above:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Numb** | **ASCII** | **ANSI** | **8859** | **UTF‑8** | **Description** |
| 32 |  |  |  |  | space |
| 33 | ! | ! | ! | ! | exclamation mark |
| 34 | " | " | " | " | quotation mark |
| 35 | # | # | # | # | number sign |
| 36 | $ | $ | $ | $ | dollar sign |
| 37 | % | % | % | % | percent sign |
| 38 | & | & | & | & | ampersand |
| 39 | ' | ' | ' | ' | apostrophe |
| 40 | ( | ( | ( | ( | left parenthesis |
| 41 | ) | ) | ) | ) | right parenthesis |
| 42 | \* | \* | \* | \* | asterisk |
| 43 | + | + | + | + | plus sign |
| 44 | , | , | , | , | comma |
| 45 | - | - | - | - | hyphen-minus |
| 46 | . | . | . | . | full stop |
| 47 | / | / | / | / | solidus |
| 48 | 0 | 0 | 0 | 0 | digit zero |
| 49 | 1 | 1 | 1 | 1 | digit one |
| 50 | 2 | 2 | 2 | 2 | digit two |
| 51 | 3 | 3 | 3 | 3 | digit three |
| 52 | 4 | 4 | 4 | 4 | digit four |
| 53 | 5 | 5 | 5 | 5 | digit five |
| 54 | 6 | 6 | 6 | 6 | digit six |
| 55 | 7 | 7 | 7 | 7 | digit seven |
| 56 | 8 | 8 | 8 | 8 | digit eight |
| 57 | 9 | 9 | 9 | 9 | digit nine |
| 58 | : | : | : | : | colon |
| 59 | ; | ; | ; | ; | semicolon |
| 60 | < | < | < | < | less than |
| 61 | = | = | = | = | equals sign |
| 62 | > | > | > | > | greater than |
| 63 | ? | ? | ? | ? | question mark |
| 64 | @ | @ | @ | @ | commercial at |
| 65 | A | A | A | A | Latin A |
| 66 | B | B | B | B | Latin B |
| 67 | C | C | C | C | Latin C |
| 68 | D | D | D | D | Latin D |
| 69 | E | E | E | E | Latin E |
| 70 | F | F | F | F | Latin F |
| 71 | G | G | G | G | Latin G |
| 72 | H | H | H | H | Latin H |
| 73 | I | I | I | I | Latin I |
| 74 | J | J | J | J | Latin J |
| 75 | K | K | K | K | Latin K |
| 76 | L | L | L | L | Latin L |
| 77 | M | M | M | M | Latin M |
| 78 | N | N | N | N | Latin N |
| 79 | O | O | O | O | Latin O |
| 80 | P | P | P | P | Latin P |
| 81 | Q | Q | Q | Q | Latin Q |
| 82 | R | R | R | R | Latin R |
| 83 | S | S | S | S | Latin S |
| 84 | T | T | T | T | Latin T |
| 85 | U | U | U | U | Latin U |
| 86 | V | V | V | V | Latin V |
| 87 | W | W | W | W | Latin W |
| 88 | X | X | X | X | Latin X |
| 89 | Y | Y | Y | Y | Latin Y |
| 90 | Z | Z | Z | Z | Latin Z |
| 91 | [ | [ | [ | [ | left square bracket |
| 92 | \ | \ | \ | \ | reverse solidus |
| 93 | ] | ] | ] | ] | right square bracket |
| 94 | ^ | ^ | ^ | ^ | circumflex accent |
| 95 | \_ | \_ | \_ | \_ | low line |
| 96 | ` | ` | ` | ` | grave accent |
| 97 | a | a | a | a | Latin small a |
| 98 | b | b | b | b | Latin small b |
| 99 | c | c | c | c | Latin small c |
| 100 | d | d | d | d | Latin small d |
| 101 | e | e | e | e | Latin small e |
| 102 | f | f | f | f | Latin small f |
| 103 | g | g | g | g | Latin small g |
| 104 | h | h | h | h | Latin small h |
| 105 | i | i | i | i | Latin small i |
| 106 | j | j | j | j | Latin small j |
| 107 | k | k | k | k | Latin small k |
| 108 | l | l | l | l | Latin small l |
| 109 | m | m | m | m | Latin small m |
| 110 | n | n | n | n | Latin small n |
| 111 | o | o | o | o | Latin small o |
| 112 | p | p | p | p | Latin small p |
| 113 | q | q | q | q | Latin small q |
| 114 | r | r | r | r | Latin small r |
| 115 | s | s | s | s | Latin small s |
| 116 | t | t | t | t | Latin small t |
| 117 | u | u | u | u | Latin small u |
| 118 | v | v | v | v | Latin small v |
| 119 | w | w | w | w | Latin small w |
| 120 | x | x | x | x | Latin small x |
| 121 | y | y | y | y | Latin small y |
| 122 | z | z | z | z | Latin small z |
| 123 | { | { | { | { | left curly bracket |
| 124 | | | | | | | | | vertical line |
| 125 | } | } | } | } | right curly bracket |
| 126 | ~ | ~ | ~ | ~ | tilde |
| 127 | DEL |  |  |  |  |
| 128 |  | € |  |  | euro sign |
| 129 |  |  |  |  | NOT USED |
| 130 |  | ‚ |  |  | single low-9 quotation mark |
| 131 |  | ƒ |  |  | Latin small f with hook |
| 132 |  | „ |  |  | double low-9 quotation mark |
| 133 |  | … |  |  | horizontal ellipsis |
| 134 |  | † |  |  | dagger |
| 135 |  | ‡ |  |  | double dagger |
| 136 |  | ˆ |  |  | modifier letter circumflex accent |
| 137 |  | ‰ |  |  | per mille sign |
| 138 |  | Š |  |  | Latin S with caron |
| 139 |  | ‹ |  |  | single left-pointing angle quotation mark |
| 140 |  | Œ |  |  | Latin capital ligature OE |
| 141 |  |  |  |  | NOT USED |
| 142 |  | Ž |  |  | Latin Z with caron |
| 143 |  |  |  |  | NOT USED |
| 144 |  |  |  |  | NOT USED |
| 145 |  | ‘ |  |  | left single quotation mark |
| 146 |  | ’ |  |  | right single quotation mark |
| 147 |  | “ |  |  | left double quotation mark |
| 148 |  | ” |  |  | right double quotation mark |
| 149 |  | • |  |  | bullet |
| 150 |  | – |  |  | en dash |
| 151 |  | — |  |  | em dash |
| 152 |  | ˜ |  |  | small tilde |
| 153 |  | ™ |  |  | trade mark sign |
| 154 |  | š |  |  | Latin small s with caron |
| 155 |  | › |  |  | single right-pointing angle quotation mark |
| 156 |  | œ |  |  | Latin small ligature oe |
| 157 |  |  |  |  | NOT USED |
| 158 |  | ž |  |  | Latin small z with caron |
| 159 |  | Ÿ |  |  | Latin Y with diaeresis |
| 160 |  |  |  |  | no-break space |
| 161 |  | ¡ | ¡ | ¡ | inverted exclamation mark |
| 162 |  | ¢ | ¢ | ¢ | cent sign |
| 163 |  | £ | £ | £ | pound sign |
| 164 |  | ¤ | ¤ | ¤ | currency sign |
| 165 |  | ¥ | ¥ | ¥ | yen sign |
| 166 |  | ¦ | ¦ | ¦ | broken bar |
| 167 |  | § | § | § | section sign |
| 168 |  | ¨ | ¨ | ¨ | diaeresis |
| 169 |  | © | © | © | copyright sign |
| 170 |  | ª | ª | ª | feminine ordinal indicator |
| 171 |  | « | « | « | left-pointing double angle quotation mark |
| 172 |  | ¬ | ¬ | ¬ | not sign |
| 173 |  | ­ | ­ | ­ | soft hyphen |
| 174 |  | ® | ® | ® | registered sign |
| 175 |  | ¯ | ¯ | ¯ | macron |
| 176 |  | ° | ° | ° | degree sign |
| 177 |  | ± | ± | ± | plus-minus sign |
| 178 |  | ² | ² | ² | superscript two |
| 179 |  | ³ | ³ | ³ | superscript three |
| 180 |  | ´ | ´ | ´ | acute accent |
| 181 |  | µ | µ | µ | micro sign |
| 182 |  | ¶ | ¶ | ¶ | pilcrow sign |
| 183 |  | · | · | · | middle dot |
| 184 |  | ¸ | ¸ | ¸ | cedilla |
| 185 |  | ¹ | ¹ | ¹ | superscript one |
| 186 |  | º | º | º | masculine ordinal indicator |
| 187 |  | » | » | » | right-pointing double angle quotation mark |
| 188 |  | ¼ | ¼ | ¼ | vulgar fraction one quarter |
| 189 |  | ½ | ½ | ½ | vulgar fraction one half |
| 190 |  | ¾ | ¾ | ¾ | vulgar fraction three quarters |
| 191 |  | ¿ | ¿ | ¿ | inverted question mark |
| 192 |  | À | À | À | Latin A with grave |
| 193 |  | Á | Á | Á | Latin A with acute |
| 194 |  | Â | Â | Â | Latin A with circumflex |
| 195 |  | Ã | Ã | Ã | Latin A with tilde |
| 196 |  | Ä | Ä | Ä | Latin A with diaeresis |
| 197 |  | Å | Å | Å | Latin A with ring above |
| 198 |  | Æ | Æ | Æ | Latin AE |
| 199 |  | Ç | Ç | Ç | Latin C with cedilla |
| 200 |  | È | È | È | Latin E with grave |
| 201 |  | É | É | É | Latin E with acute |
| 202 |  | Ê | Ê | Ê | Latin E with circumflex |
| 203 |  | Ë | Ë | Ë | Latin E with diaeresis |
| 204 |  | Ì | Ì | Ì | Latin I with grave |
| 205 |  | Í | Í | Í | Latin I with acute |
| 206 |  | Î | Î | Î | Latin I with circumflex |
| 207 |  | Ï | Ï | Ï | Latin I with diaeresis |
| 208 |  | Ð | Ð | Ð | Latin Eth |
| 209 |  | Ñ | Ñ | Ñ | Latin N with tilde |
| 210 |  | Ò | Ò | Ò | Latin O with grave |
| 211 |  | Ó | Ó | Ó | Latin O with acute |
| 212 |  | Ô | Ô | Ô | Latin O with circumflex |
| 213 |  | Õ | Õ | Õ | Latin O with tilde |
| 214 |  | Ö | Ö | Ö | Latin O with diaeresis |
| 215 |  | × | × | × | multiplication sign |
| 216 |  | Ø | Ø | Ø | Latin O with stroke |
| 217 |  | Ù | Ù | Ù | Latin U with grave |
| 218 |  | Ú | Ú | Ú | Latin U with acute |
| 219 |  | Û | Û | Û | Latin U with circumflex |
| 220 |  | Ü | Ü | Ü | Latin U with diaeresis |
| 221 |  | Ý | Ý | Ý | Latin Y with acute |
| 222 |  | Þ | Þ | Þ | Latin Thorn |
| 223 |  | ß | ß | ß | Latin small sharp s |
| 224 |  | à | à | à | Latin small a with grave |
| 225 |  | á | á | á | Latin small a with acute |
| 226 |  | â | â | â | Latin small a with circumflex |
| 227 |  | ã | ã | ã | Latin small a with tilde |
| 228 |  | ä | ä | ä | Latin small a with diaeresis |
| 229 |  | å | å | å | Latin small a with ring above |
| 230 |  | æ | æ | æ | Latin small ae |
| 231 |  | ç | ç | ç | Latin small c with cedilla |
| 232 |  | è | è | è | Latin small e with grave |
| 233 |  | é | é | é | Latin small e with acute |
| 234 |  | ê | ê | ê | Latin small e with circumflex |
| 235 |  | ë | ë | ë | Latin small e with diaeresis |
| 236 |  | ì | ì | ì | Latin small i with grave |
| 237 |  | í | í | í | Latin small i with acute |
| 238 |  | î | î | î | Latin small i with circumflex |
| 239 |  | ï | ï | ï | Latin small i with diaeresis |
| 240 |  | ð | ð | ð | Latin small eth |
| 241 |  | ñ | ñ | ñ | Latin small n with tilde |
| 242 |  | ò | ò | ò | Latin small o with grave |
| 243 |  | ó | ó | ó | Latin small o with acute |
| 244 |  | ô | ô | ô | Latin small o with circumflex |
| 245 |  | õ | õ | õ | Latin small o with tilde |
| 246 |  | ö | ö | ö | Latin small o with diaeresis |
| 247 |  | ÷ | ÷ | ÷ | division sign |
| 248 |  | ø | ø | ø | Latin small o with stroke |
| 249 |  | ù | ù | ù | Latin small u with grave |
| 250 |  | ú | ú | ú | Latin small u with acute |
| 251 |  | û | û | û | Latin small with circumflex |
| 252 |  | ü | ü | ü | Latin small u with diaeresis |
| 253 |  | ý | ý | ý | Latin small y with acute |
| 254 |  | þ | þ | þ | Latin small thorn |
| 255 |  | ÿ | ÿ | ÿ | Latin small y with diaeresis |

# **HTML**Uniform Resource Locators

A URL is another word for a web address.

A URL can be composed of words (e.g. w3schools.com), or an Internet Protocol (IP) address (e.g. 192.68.20.50).

Most people enter the name when surfing, because names are easier to remember than numbers.

## **URL - Uniform Resource Locator**

Web browsers request pages from web servers by using a URL.

A Uniform Resource Locator (URL) is used to address a document (or other data) on the web.

scheme://prefix.domain:port/path/filename

Explanation:

* **scheme** - defines the **type** of Internet service (most common is **http or https**)
* **prefix** - defines a domain **prefix** (default for http is **www**)
* **domain** - defines the Internet **domain name**(like w3schools.com)
* **port** - defines the **port number**at the host (default for http is **80**)
* **path** - defines a **path** at the server (If omitted: the root directory of the site)
* **filename** - defines the name of a document or resource

## **Common URL Schemes**

The table below lists some common schemes:

|  |  |  |
| --- | --- | --- |
| **Scheme** | **Short for** | **Used for** |
| http | HyperText Transfer Protocol | Common web pages. Not encrypted |
| https | Secure HyperText Transfer Protocol | Secure web pages. Encrypted |
| ftp | File Transfer Protocol | Downloading or uploading files |
| file |  | A file on your computer |

## **URL Encoding**

URLs can only be sent over the Internet using the [ASCII character-set](https://www.w3schools.com/charsets/ref_html_ascii.asp). If a URL contains characters outside the ASCII set, the URL has to be converted.

URL encoding converts non-ASCII characters into a format that can be transmitted over the Internet.

URL encoding replaces non-ASCII characters with a "%" followed by hexadecimal digits.

URLs cannot contain spaces. URL encoding normally replaces a space with a plus (+) sign, or %20.

## **Try It Yourself**

Top of Form

Bottom of Form

If you click "Submit", the browser will URL encode the input before it is sent to the server.

A page at the server will display the received input.

Try some other input and click Submit again.

## **ASCII Encoding Examples**

Your browser will encode input, according to the character-set used in your page.

The default character-set in HTML5 is UTF-8.

|  |  |  |
| --- | --- | --- |
| **Character** | **From Windows-1252** | **From UTF-8** |
| € | %80 | %E2%82%AC |
| £ | %A3 | %C2%A3 |
| © | %A9 | %C2%A9 |
| ® | %AE | %C2%AE |
| À | %C0 | %C3%80 |
| Á | %C1 | %C3%81 |
| Â | %C2 | %C3%82 |
| Ã | %C3 | %C3%83 |
| Ä | %C4 | %C3%84 |
| Å | %C5 | %C3%85 |

# **HTML** Versus XHTML

XHTML is a stricter, more XML-based version of HTML.

## **What is XHTML?**

* XHTML stands for E**X**tensible **H**yper**T**ext **M**arkup **L**anguage
* XHTML is a stricter, more XML-based version of HTML
* XHTML is HTML defined as an XML application
* XHTML is supported by all major browsers

## **Why XHTML?**

XML is a markup language where all documents must be marked up correctly (be "well-formed").

XHTML was developed to make HTML more extensible and flexible to work with other data formats (such as XML). In addition, browsers ignore errors in HTML pages, and try to display the website even if it has some errors in the markup. So XHTML comes with a much stricter error handling.

If you want to study XML, please read our [XML Tutorial](https://www.w3schools.com/xml/default.asp).

## **The Most Important Differences from HTML**

* <!DOCTYPE> is **mandatory**
* The xmlns attribute in <html> is **mandatory**
* <html>, <head>, <title>, and <body> are **mandatory**
* Elements must always be **properly nested**
* Elements must always be **closed**
* Elements must always be in **lowercase**
* Attribute names must always be in **lowercase**
* Attribute values must always be **quoted**
* Attribute minimization is **forbidden**

## **XHTML - <!DOCTYPE ....> Is Mandatory**

An XHTML document must have an XHTML <!DOCTYPE> declaration.

The <html>, <head>, <title>, and <body> elements must also be present, and the xmlns attribute in <html> must specify the xml namespace for the document.

### Example

Here is an XHTML document with a minimum of required tags:

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"  
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml">  
<head>  
  <title>Title of document</title>  
</head>  
<body>  
  
  some content here...  
  
</body>  
</html>

## **XHTML Elements Must be Properly Nested**

In XHTML, elements must always be properly nested within each other, like this:

### Correct:

<b><i>Some text</i></b>

### Wrong:

<b><i>Some text</b></i>

## **XHTML Elements Must Always be Closed**

In XHTML, elements must always be closed, like this:

### Correct:

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

### Wrong:

<p>This is a paragraph  
<p>This is another paragraph

## **XHTML Empty Elements Must Always be Closed**

In XHTML, empty elements must always be closed, like this:

### Correct:

A break: <br />  
A horizontal rule: <hr />  
An image: <img src="happy.gif" alt="Happy face" />

### Wrong:

A break: <br>  
A horizontal rule: <hr>  
An image: <img src="happy.gif" alt="Happy face">

## **XHTML Elements Must be in Lowercase**

In XHTML, element names must always be in lowercase, like this:

### Correct:

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

### Wrong:

<BODY>  
<P>This is a paragraph</P>  
</BODY>

## **XHTML Attribute Names Must be in Lowercase**

In XHTML, attribute names must always be in lowercase, like this:

### Correct:

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

### Wrong:

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

## **XHTML Attribute Values Must be Quoted**

In XHTML, attribute values must always be quoted, like this:

### Correct:

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

### Wrong:

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

## **XHTML Attribute Minimization is Forbidden**

In XHTML, attribute minimization is forbidden:

### Correct:

<input type="checkbox" name="vehicle" value="car" checked="checked" />  
<input type="text" name="lastname" disabled="disabled" />

### Wrong:

<input type="checkbox" name="vehicle" value="car" checked />  
<input type="text" name="lastname" disabled />

# **HTML Canvas Graphics:**

The HTML <canvas> element is used to draw graphics on a web page.

The graphic to the left is created with <canvas>. It shows four elements: a red rectangle, a gradient rectangle, a multicolor rectangle, and a multicolor text.

## **What is HTML Canvas?**

The HTML <canvas> element is used to draw graphics, on the fly, via JavaScript.

The <canvas> element is only a container for graphics. You must use JavaScript to actually draw the graphics.

Canvas has several methods for drawing paths, boxes, circles, text, and adding images.

## **Browser Support**

The numbers in the table specify the first browser version that fully supports the <canvas> element.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Element |  |  |  |  |  |
| <canvas> | 4.0 | 9.0 | 2.0 | 3.1 | 9.0 |

## **Canvas Examples**

A canvas is a rectangular area on an HTML page. By default, a canvas has no border and no content.

The markup looks like this:

<canvas id="myCanvas" width="200" height="100"></canvas>

**Note:** Always specify an id attribute (to be referred to in a script), and a width and height attribute to define the size of the canvas. To add a border, use the style attribute.

Here is an example of a basic, empty canvas:

### Example

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #000000;">  
</canvas>

ADVERTISEMENT

## **Add a JavaScript:**

After creating the rectangular canvas area, you must add a JavaScript to do the drawing.

Here are some examples:

### Draw a Line

### Example

<script>  
var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.moveTo(0, 0);  
ctx.lineTo(200, 100);  
ctx.stroke();  
</script>

<!DOCTYPE html>

<html>

<body>

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">

Your browser does not support the HTML canvas tag.</canvas>

<script>

var c = document.getElementById("myCanvas");

var ctx = c.getContext("2d");

ctx.moveTo(0,0);

ctx.lineTo(200,100);

ctx.stroke();

</script>

</body>

</html>

### Example

<script>  
var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.beginPath();  
ctx.arc(95, 50, 40, 0, 2 \* Math.PI);  
ctx.stroke();  
</script>

<!DOCTYPE html>

<html>

<body>

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">

Your browser does not support the HTML canvas tag.</canvas>

<script>

var c = document.getElementById("myCanvas");

var ctx = c.getContext("2d");

ctx.beginPath();

ctx.arc(95,50,40,0,2\*Math.PI);

ctx.stroke();

</script>

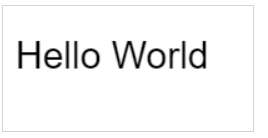
</body>

</html>

### Draw a Text

### Example

<script>  
var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.font = "30px Arial";  
ctx.fillText("Hello World", 10, 50);  
</script>



### Example

<script>  
var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.font = "30px Arial";  
ctx.strokeText("Hello World", 10, 50);  
</script>

Output:



<!DOCTYPE html>

<html>

<body>

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">

Your browser does not support the HTML canvas tag.</canvas>

<script>

var c = document.getElementById("myCanvas");

var ctx = c.getContext("2d");

// Create gradient

var grd = ctx.createLinearGradient(0,0,200,0);

grd.addColorStop(0,"red");

grd.addColorStop(1,"white");

// Fill with gradient

ctx.fillStyle = grd;

ctx.fillRect(10,10,150,80);

</script>

</body>

</html>

Output:



### Example

<script>  
var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
  
// Create gradient  
var grd = ctx.createRadialGradient(75, 50, 5, 90, 60, 100);  
grd.addColorStop(0, "red");  
grd.addColorStop(1, "white");  
  
// Fill with gradient  
ctx.fillStyle = grd;  
ctx.fillRect(10, 10, 150, 80);  
</script>



### Example

<script>  
var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
  
// Create gradient  
var grd = ctx.createRadialGradient(75, 50, 5, 90, 60, 100);  
grd.addColorStop(0, "red");  
grd.addColorStop(1, "white");  
  
// Fill with gradient  
ctx.fillStyle = grd;  
ctx.fillRect(10, 10, 150, 80);  
</script>

Output:



### Draw Image

<script>  
var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
var img = document.getElementById("scream");  
ctx.drawImage(img, 10, 10);  
</script>

Program:

<!DOCTYPE html>

<html>

<body>

<p>Image to use:</p>

<img id="scream" src="img\_the\_scream.jpg" alt="The Scream" width="220" height="277">

<p>Canvas to fill:</p>

<canvas id="myCanvas" width="250" height="300"

style="border:1px solid #d3d3d3;">

Your browser does not support the HTML canvas tag.</canvas>

<p><button onclick="myCanvas()">Try it</button></p>

<script>

function myCanvas() {

var c = document.getElementById("myCanvas");

var ctx = c.getContext("2d");

var img = document.getElementById("scream");

ctx.drawImage(img,10,10);

}

</script>

</body>

</html>

Output:



# HTML SVG Graphics

SVG defines vector-based graphics in XML format.

## **What is SVG?**

* SVG stands for Scalable Vector Graphics
* SVG is used to define graphics for the Web

## **The HTML <svg> Element**

The HTML <svg> element is a container for SVG graphics.

SVG has several methods for drawing paths, boxes, circles, text, and graphic images.

**EXAMPLE**

<!DOCTYPE html>

<html>

<body>

<svg width="100" height="100">

<circle cx="50" cy="50" r="40"

stroke="green" stroke-width="4" fill="yellow" />

Sorry, your browser does not support inline SVG.

</svg>

</body>

</html>

## **Browser Support**

The numbers in the table specify the first browser version that fully supports the <svg> element.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Element |  |  |  |  |  |
| <svg> | 4.0 | 9.0 | 3.0 | 3.2 | 10.1 |

## **SVG Rectangle**

### Example

## **SVG Rectangle**

### Example

## SVG Rounded Rectangle

### Example

# **<!DOCTYPE html>**

# **<html>**

# **<body>**

# **<svg width="400" height="180">**

# **<rect x="50" y="20" rx="20" ry="20" width="150" height="150"**

# **style="fill:red;stroke:black;stroke-width:5;opacity:0.5" />**

# **Sorry, your browser does not support inline SVG.**

# **</svg>**

# **</body>**

# **</html>**

## **SVG Star**

### Example

<!DOCTYPE html>

<html>

<body>

<svg width="300" height="200">

<polygon points="100,10 40,198 190,78 10,78 160,198"

style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;" />

Sorry, your browser does not support inline SVG.

</svg>

</body>

</html>

## **SVG Logo**

SVG

### Example

<!DOCTYPE html>

<html>

<body>

<svg height="130" width="500">

<defs>

<linearGradient id="grad1" x1="0%" y1="0%" x2="100%" y2="0%">

<stop offset="0%"

style="stop-color:rgb(255,255,0);stop-opacity:1" />

<stop offset="100%"

style="stop-color:rgb(255,0,0);stop-opacity:1" />

</linearGradient>

</defs>

<ellipse cx="100" cy="70" rx="85" ry="55" fill="url(#grad1)" />

<text fill="#ffffff" font-size="45" font-family="Verdana"

x="50" y="86">SVG</text>

Sorry, your browser does not support inline SVG.

</svg>

</body>

</html>

## **Differences Between SVG and Canvas**

SVG is a language for describing 2D graphics in XML.

Canvas draws 2D graphics, on the fly (with JavaScript).

SVG is XML based, which means that every element is available within the SVG DOM. You can attach JavaScript event handlers for an element.

In SVG, each drawn shape is remembered as an object. If attributes of an SVG object are changed, the browser can automatically re-render the shape.

Canvas is rendered pixel by pixel. In canvas, once the graphic is drawn, it is forgotten by the browser. If its position should be changed, the entire scene needs to be redrawn, including any objects that might have been covered by the graphic.

## **Comparison of Canvas and SVG**

The table below shows some important differences between Canvas and SVG:

|  |  |
| --- | --- |
| **Canvas** | **SVG** |
| * Resolution dependent * No support for event handlers * Poor text rendering capabilities * You can save the resulting image as .png or .jpg * Well suited for graphic-intensive games | * Resolution independent * Support for event handlers * Best suited for applications with large rendering areas (Google Maps) * Slow rendering if complex (anything that uses the DOM a lot will be slow) * Not suited for game applications |

# HTML Multimedia

Multimedia on the web is sound, music, videos, movies, and animations.

## **What is Multimedia?**

Multimedia comes in many different formats. It can be almost anything you can hear or see, like images, music, sound, videos, records, films, animations, and more.

Web pages often contain multimedia elements of different types and formats.

## **Browser Support**

The first web browsers had support for text only, limited to a single font in a single color.

Later came browsers with support for colors, fonts, images, and multimedia!

## **Multimedia Formats**

Multimedia elements (like audio or video) are stored in media files.

The most common way to discover the type of a file, is to look at the file extension.

Multimedia files have formats and different extensions like: .wav, .mp3, .mp4, .mpg, .wmv, and .avi.

## **Common Video Formats**

|  |  |
| --- | --- |
| Videoformats | There are many video formats out there.  The MP4, WebM, and Ogg formats are supported by HTML.  The MP4 format is recommended by YouTube. |

|  |  |  |
| --- | --- | --- |
| **Format** | **File** | **Description** |
| MPEG | .mpg .mpeg | MPEG. Developed by the Moving Pictures Expert Group. The first popular video format on the web. Not supported anymore in HTML. |
| AVI | .avi | AVI (Audio Video Interleave). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers. |
| WMV | .wmv | WMV (Windows Media Video). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers. |
| QuickTime | .mov | QuickTime. Developed by Apple. Commonly used in video cameras and TV hardware. Plays well on Apple computers, but not in web browsers. |
| RealVideo | .rm .ram | RealVideo. Developed by Real Media to allow video streaming with low bandwidths. Does not play in web browsers. |
| Flash | .swf .flv | Flash. Developed by Macromedia. Often requires an extra component (plug-in) to play in web browsers. |
| Ogg | .ogg | Theora Ogg. Developed by the Xiph.Org Foundation. Supported by HTML. |
| WebM | .webm | WebM. Developed by Mozilla, Opera, Adobe, and Google. Supported by HTML. |
| MPEG-4 or MP4 | .mp4 | MP4. Developed by the Moving Pictures Expert Group. Commonly used in video cameras and TV hardware. Supported by all browsers and  recommended by YouTube. |

**Note:** Only MP4, WebM, and Ogg video are supported by the HTML standard.

## **Common Audio Formats**

MP3 is the best format for compressed recorded music. The term MP3 has become synonymous with digital music.

If your website is about recorded music, MP3 is the choice.

|  |  |  |
| --- | --- | --- |
| **Format** | **File** | **Description** |
| MIDI | .mid .midi | MIDI (Musical Instrument Digital Interface). Main format for all electronic music devices like synthesizers and PC sound cards. MIDI files do not contain sound, but digital notes that can be played by electronics. Plays well on all computers and music hardware, but not in web browsers. |
| RealAudio | .rm .ram | RealAudio. Developed by Real Media to allow streaming of audio with low bandwidths. Does not play in web browsers. |
| WMA | .wma | WMA (Windows Media Audio). Developed by Microsoft. Plays well on Windows computers, but not in web browsers. |
| AAC | .aac | AAC (Advanced Audio Coding). Developed by Apple as the default format for iTunes. Plays well on Apple computers, but not in web browsers. |
| WAV | .wav | WAV. Developed by IBM and Microsoft. Plays well on Windows, Macintosh, and Linux operating systems. Supported by HTML. |
| Ogg | .ogg | Ogg. Developed by the Xiph.Org Foundation. Supported by HTML. |
| MP3 | .mp3 | MP3 files are actually the sound part of MPEG files. MP3 is the most popular format for music players. Combines good compression (small files) with high quality. Supported by all browsers. |
| MP4 | .mp4 | MP4 is a video format, but can also be used for audio. Supported by all browsers. |

**Note:** Only MP3, WAV, and Ogg audio are supported by the HTML standard.

# HTML Video

The HTML <video> element is used to show a video on a web page.

### Example

Courtesy of [Big Buck Bunny](https://www.bigbuckbunny.org/):

<!DOCTYPE html>

<html>

<body>

<video width="400" controls>

<source src="mov\_bbb.mp4" type="video/mp4">

<source src="mov\_bbb.ogg" type="video/ogg">

Your browser does not support HTML video.

</video>

<p>

Video courtesy of

<a href="https://www.bigbuckbunny.org/" target="\_blank">Big Buck Bunny</a>.

</p>

</body>

</html>

**OUTPUT**

Video courtesy of [Big Buck Bunny](https://www.bigbuckbunny.org/).

## **The HTML <video> Element**

To show a video in HTML, use the <video> element:

<!DOCTYPE html>

<html>

<body>

<video width="320" height="240" controls>

<source src="movie.mp4" type="video/mp4">

<source src="movie.ogg" type="video/ogg">

Your browser does not support the video tag.

</video>

</body>

</html>

## **How it Works**

The controls attribute adds video controls, like play, pause, and volume.

It is a good idea to always include width and height attributes. If height and width are not set, the page might flicker while the video loads.

The <source> element allows you to specify alternative video files which the browser may choose from. The browser will use the first recognized format.

The text between the <video> and </video> tags will only be displayed in browsers that do not support the <video> element.

## **HTML <video> Autoplay**

To start a video automatically, use the autoplay attribute:

### Example

<!DOCTYPE html>

<html>

<body>

<video width="320" height="240" autoplay>

<source src="movie.mp4" type="video/mp4">

<source src="movie.ogg" type="video/ogg">

Your browser does not support the video tag.

</video>

</body>

</html>

**Note:** Chromium browsers do not allow autoplay in most cases. However, muted autoplay is always allowed.

Add muted after autoplay to let your video start playing automatically (but muted):

### Example

<!DOCTYPE html>

<html>

<body>

<video width="320" height="240" autoplay muted>

<source src="movie.mp4" type="video/mp4">

<source src="movie.ogg" type="video/ogg">

Your browser does not support the video tag.

</video>

</body>

</html>

## **Browser Support**

The numbers in the table specify the first browser version that fully supports the <video> element.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Element |  |  |  |  |  |
| <video> | 4.0 | 9.0 | 3.5 | 4.0 | 10.5 |

## **HTML Video Formats**

There are three supported video formats: MP4, WebM, and Ogg. The browser support for the different formats is:

|  |  |  |  |
| --- | --- | --- | --- |
| **Browser** | **MP4** | **WebM** | **Ogg** |
| Edge | YES | YES | YES |
| Chrome | YES | YES | YES |
| Firefox | YES | YES | YES |
| Safari | YES | YES | NO |
| Opera | YES | YES | YES |

## **HTML Video - Media Types**

|  |  |
| --- | --- |
| **File Format** | **Media Type** |
| MP4 | video/mp4 |
| WebM | video/webm |
| Ogg | video/ogg |

## **HTML Video - Methods, Properties, and Events**

The HTML DOM defines methods, properties, and events for the <video> element.

This allows you to load, play, and pause videos, as well as setting duration and volume.

There are also DOM events that can notify you when a video begins to play, is paused, etc.

### Example: Using JavaScript

For a full DOM reference, go to our [HTML Audio/Video DOM Reference](https://www.w3schools.com/tags/ref_av_dom.asp).

## **HTML Video Tags**

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<video>](https://www.w3schools.com/tags/tag_video.asp) | Defines a video or movie |
| [<source>](https://www.w3schools.com/tags/tag_source.asp) | Defines multiple media resources for media elements, such as <video> and <audio> |
| [<track>](https://www.w3schools.com/tags/tag_track.asp) | Defines text tracks in media players |

# HTML Audio

The HTML <audio> element is used to play an audio file on a web page.

## **The HTML <audio> Element**

To play an audio file in HTML, use the <audio> element:

### Example

# **<!DOCTYPE html>**

# **<html>**

# **<body>**

# **<audio controls>**

# **<source src="horse.ogg" type="audio/ogg">**

# **<source src="horse.mp3" type="audio/mpeg">**

# **Your browser does not support the audio element.**

# **</audio>**

# **</body>**

# **</html>**

## **HTML Audio - How It Works**

The controls attribute adds audio controls, like play, pause, and volume.

The <source> element allows you to specify alternative audio files which the browser may choose from. The browser will use the first recognized format.

The text between the <audio> and </audio> tags will only be displayed in browsers that do not support the <audio> element.

## **HTML <audio> Autoplay**

To start an audio file automatically, use the autoplay attribute:

### Example

## **HTML Audio - How It Works**

The controls attribute adds audio controls, like play, pause, and volume.

The <source> element allows you to specify alternative audio files which the browser may choose from. The browser will use the first recognized format.

The text between the <audio> and </audio> tags will only be displayed in browsers that do not support the <audio> element.

## **HTML <audio> Autoplay**

To start an audio file automatically, use the autoplay attribute:

### Example

<!DOCTYPE html>

<html>

<body>

<audio controls autoplay>

<source src="horse.ogg" type="audio/ogg">

<source src="horse.mp3" type="audio/mpeg">

Your browser does not support the audio element.

</audio>

</body>

</html>

## **Browser Support**

The numbers in the table specify the first browser version that fully supports the <audio> element.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Element |  |  |  |  |  |
| <audio> | 4.0 | 9.0 | 3.5 | 4.0 | 10.5 |

## **HTML Audio Formats**

There are three supported audio formats: MP3, WAV, and OGG. The browser support for the different formats is:

|  |  |  |  |
| --- | --- | --- | --- |
| **Browser** | **MP3** | **WAV** | **OGG** |
| Edge/IE | YES | YES\* | YES\* |
| Chrome | YES | YES | YES |
| Firefox | YES | YES | YES |
| Safari | YES | YES | NO |
| Opera | YES | YES | YES |

\*From Edge 79

## **HTML Audio - Media Types**

|  |  |
| --- | --- |
| **File Format** | **Media Type** |
| MP3 | audio/mpeg |
| OGG | audio/ogg |
| WAV | audio/wav |

## HTML Audio - Methods, Properties, and Events

The HTML DOM defines methods, properties, and events for the <audio> element.

This allows you to load, play, and pause audios, as well as set duration and volume.

There are also DOM events that can notify you when an audio begins to play, is paused, etc.

## HTML Audio Tags

|  |  |
| --- | --- |
| Tag | Description |
| [<audio>](https://www.w3schools.com/tags/tag_audio.asp) | Defines sound content |
| [<source>](https://www.w3schools.com/tags/tag_source.asp) | Defines multiple media resources for media elements, such as <video> and <audio> |

# **HTML Plug-ins**

Plug-ins are computer programs that extend the standard functionality of the browser.

## **Plug-ins**

Plug-ins were designed to be used for many different purposes:

* To run Java applets
* To run Microsoft ActiveX controls
* To display Flash movies
* To display maps
* To scan for viruses
* To verify a bank id

**Warning !**

Most browsers no longer support Java Applets and Plug-ins.

ActiveX controls are no longer supported in any browsers.

The support for Shockwave Flash has also been turned off in modern browsers.

## **The <object> Element**

The <object> element is supported by all browsers.

The <object> element defines an embedded object within an HTML document.

It was designed to embed plug-ins (like Java applets, PDF readers, and Flash Players) in web pages, but can also be used to include HTML in HTML:

### Example

<!DOCTYPE html>

<html>

<body>

<object width="100%" height="500px" data="snippet.html"></object>

</body>

</html>

**OUTPUT**

|  |  |  |
| --- | --- | --- |
| Alfreds Futterkiste | Berlin | Germany |
| Berglunds snabbköp | Luleå | Sweden |
| Centro comercial Moctezuma | México D.F. | Mexico |
| Ernst Handel | Graz | Austria |
| FISSA Fabrica Inter. Salchichas S.A. | Madrid | Spain |
| Galería del gastrónomo | Barcelona | Spain |
| Island Trading | Cowes | UK |
| Königlich Essen | Brandenburg | Germany |
| Laughing Bacchus Wine Cellars | Vancouver | Canada |
| Magazzini Alimentari Riuniti | Bergamo | Italy |
| North/South | London | UK |
| Paris spécialités | Paris | France |
| Rattlesnake Canyon Grocery | Albuquerque | USA |
| Simons bistro | København | Denmark |

**EXAMPLE**

<!DOCTYPE html>

<html>

<body>

<object data="audi.jpeg"></object>

</body>

</html>

## **The <embed> Element**

The <embed> element is supported in all major browsers.

The <embed> element also defines an embedded object within an HTML document.

Web browsers have supported the <embed> element for a long time. However, it has not been a part of the HTML specification before HTML5.

### Example

<!DOCTYPE html>

<html>

<body>

<embed src="audi.jpeg">

</body>

</html>

Note that the <embed> element does not have a closing tag. It can not contain alternative text.

The <embed> element can also be used to include HTML in HTML:

**Example**

### <!DOCTYPE html>

### <html>

### <body>

### <embed width="100%" height="500px" src="snippet.html">

### </body>

### </html>

### OUTPUT

|  |  |  |
| --- | --- | --- |
| Alfreds Futterkiste | Berlin | Germany |
| Berglunds snabbköp | Luleå | Sweden |
| Centro comercial Moctezuma | México D.F. | Mexico |
| Ernst Handel | Graz | Austria |
| FISSA Fabrica Inter. Salchichas S.A. | Madrid | Spain |
| Galería del gastrónomo | Barcelona | Spain |
| Island Trading | Cowes | UK |
| Königlich Essen | Brandenburg | Germany |
| Laughing Bacchus Wine Cellars | Vancouver | Canada |
| Magazzini Alimentari Riuniti | Bergamo | Italy |
| North/South | London | UK |
| Paris spécialités | Paris | France |
| Rattlesnake Canyon Grocery | Albuquerque | USA |
| Simons bistro | København | Denmark |
| The Big Cheese | Portland | USA |
| Vaffeljernet | Århus | Denmark |
| Wolski Zajazd | Warszawa | Poland |

# HTML YouTube Videos

The easiest way to play videos in HTML, is to use YouTube.

## **Struggling with Video Formats?**

Converting videos to different formats can be difficult and time-consuming.

An easier solution is to let YouTube play the videos in your web page.

## **YouTube Video Id**

YouTube will display an id (like tgbNymZ7vqY), when you save (or play) a video.

You can use this id, and refer to your video in the HTML code.

## **Playing a YouTube Video in HTML**

To play your video on a web page, do the following:

* Upload the video to YouTube
* Take a note of the video id
* Define an <iframe> element in your web page
* Let the src attribute point to the video URL
* Use the width and height attributes to specify the dimension of the player
* Add any other parameters to the URL (see below)

### Example

### <!DOCTYPE html>

### <html>

### <body>

### <iframe width="420" height="345" src="https://www.youtube.com/embed/tgbNymZ7vqY">

### </iframe>

### </body>

## **YouTube Autoplay + Mute**

You can let your video start playing automatically when a user visits the page, by adding autoplay=1 to the YouTube URL. However, automatically starting a video is annoying for your visitors!

**Note:** Chromium browsers do not allow autoplay in most cases. However, muted autoplay is always allowed.

Add mute=1 after autoplay=1 to let your video start playing automatically (but muted).

### YouTube - Autoplay + Muted

## **YouTube Autoplay + Mute**

You can let your video start playing automatically when a user visits the page, by adding autoplay=1 to the YouTube URL. However, automatically starting a video is annoying for your visitors!

**Note:** Chromium browsers do not allow autoplay in most cases. However, muted autoplay is always allowed.

Add mute=1 after autoplay=1 to let your video start playing automatically (but muted).

### YouTube - Autoplay + Muted

## **YouTube Playlist**

A comma separated list of videos to play (in addition to the original URL).

## **YouTube Loop**

Add loop=1 to let your video loop forever.

Value 0 (default): The video will play only once.

Value 1: The video will loop (forever).

### YouTube - Loop

## **YouTube Controls**

Add controls=0 to not display controls in the video player.

Value 0: Player controls does not display.

Value 1 (default): Player controls display.

### YouTube - Controls

<iframe width="420" height="315"  
src="https://www.youtube.com/embed/tgbNymZ7vqY?controls=0">  
</iframe>

**OUTPUT**

<!DOCTYPE html>

<html>

<body>

<iframe width="420" height="345" src="https://www.youtube.com/embed/tgbNymZ7vqY?controls=0">

</iframe>

</body>

</html>

# **HTML**Geolocation API

The HTML Geolocation API is used to locate a user's position.

## **Locate the User's Position**

The HTML Geolocation API is used to get the geographical position of a user.

Since this can compromise privacy, the position is not available unless the user approves it.

Try It

**Note:**Geolocation is most accurate for devices with GPS, like smartphones.

## **Browser Support**

The numbers in the table specify the first browser version that fully supports Geolocation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| API |  |  |  |  |  |
| Geolocation | 5.0 - 49.0 (http) 50.0 (https) | 9.0 | 3.5 | 5.0 | 16.0 |

**Note:** As of Chrome 50, the Geolocation API will only work on secure contexts such as HTTPS. If your site is hosted on an non-secure origin (such as HTTP) the requests to get the users location will no longer function.

## **Using HTML Geolocation**

The getCurrentPosition() method is used to return the user's position.

The example below returns the latitude and longitude of the user's position:

### Example

<script>  
const x = document.getElementById("demo");  
  
function getLocation() {  
  if (navigator.geolocation) {  
    navigator.geolocation.getCurrentPosition(showPosition);  
  } else {  
    x.innerHTML = "Geolocation is not supported by this browser.";  
  }  
}  
  
function showPosition(position) {  
  x.innerHTML = "Latitude: " + position.coords.latitude +  
  "<br>Longitude: " + position.coords.longitude;  
}  
</script>

Example explained:

* Check if Geolocation is supported
* If supported, run the getCurrentPosition() method. If not, display a message to the user
* If the getCurrentPosition() method is successful, it returns a coordinates object to the function specified in the parameter (showPosition)
* The showPosition() function outputs the Latitude and Longitude

The example above is a very basic Geolocation script, with no error handling.

## **Handling Errors and Rejections**

The second parameter of the getCurrentPosition() method is used to handle errors. It specifies a function to run if it fails to get the user's location:

### Example

function showError(error) {  
  switch(error.code) {  
    case error.PERMISSION\_DENIED:  
      x.innerHTML = "User denied the request for Geolocation."  
      break;  
    case error.POSITION\_UNAVAILABLE:  
      x.innerHTML = "Location information is unavailable."  
      break;  
    case error.TIMEOUT:  
      x.innerHTML = "The request to get user location timed out."  
      break;  
    case error.UNKNOWN\_ERROR:  
      x.innerHTML = "An unknown error occurred."  
      break;  
  }  
}

## **Location-specific Information**

This page has demonstrated how to show a user's position on a map.

Geolocation is also very useful for location-specific information, like:

* Up-to-date local information
* Showing Points-of-interest near the user

|  |  |
| --- | --- |
| **Property** | **Returns** |
| coords.latitude | The latitude as a decimal number (always returned) |
| coords.longitude | The longitude as a decimal number (always returned) |
| coords.accuracy | The accuracy of position (always returned) |
| coords.altitude | The altitude in meters above the mean sea level (returned if available) |
| coords.altitudeAccuracy | The altitude accuracy of position (returned if available) |
| coords.heading | The heading as degrees clockwise from North (returned if available) |
| coords.speed | The speed in meters per second (returned if available) |
| timestamp | The date/time of the response (returned if available) |

* Turn-by-turn navigation (GPS)

## **The getCurrentPosition() Method - Return Data**

The getCurrentPosition() method returns an object on success. The latitude, longitude and accuracy properties are always returned. The other properties are returned if available:

## **Geolocation Object - Other interesting Methods**

The Geolocation object also has other interesting methods:

* watchPosition() - Returns the current position of the user and continues to return updated position as the user moves (like the GPS in a car).
* clearWatch() - Stops the watchPosition() method.

The example below shows the watchPosition() method. You need an accurate GPS device to test this (like smartphone):

### Example

<script>  
const x = document.getElementById("demo");  
  
function getLocation() {  
  if (navigator.geolocation) {  
    navigator.geolocation.watchPosition(showPosition);  
  } else {  
    x.innerHTML = "Geolocation is not supported by this browser.";  
  }  
}  
function showPosition(position) {  
  x.innerHTML = "Latitude: " + position.coords.latitude +  
  "<br>Longitude: " + position.coords.longitude;  
}  
</script>

# **HTML**Drag and Drop API

In HTML, any element can be dragged and dropped.

## **Example**

W3Schools

Drag the W3Schools image into the rectangle.

## **Drag and Drop**

Drag and drop is a very common feature. It is when you "grab" an object and drag it to a different location.

## **Browser Support**

The numbers in the table specify the first browser version that fully supports Drag and Drop.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| API |  |  |  |  |  |
| Drag and Drop | 4.0 | 9.0 | 3.5 | 6.0 | 12.0 |

## **HTML Drag and Drop Example**

The example below is a simple drag and drop example:

### Example

<!DOCTYPE HTML>  
<html>  
<head>  
<script>  
function allowDrop(ev) {  
  ev.preventDefault();  
}  
  
function drag(ev) {  
  ev.dataTransfer.setData("text", ev.target.id);  
}  
  
function drop(ev) {  
  ev.preventDefault();  
  var data = ev.dataTransfer.getData("text");  
  ev.target.appendChild(document.getElementById(data));  
}  
</script>  
</head>  
<body>  
<div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)"></div>  
<img id="drag1" src="img\_logo.gif" draggable="true" ondragstart="drag(event)" width="336" height="69">  
</body>  
</html>

It might seem complicated, but lets go through all the different parts of a drag and drop event.

## **Make an Element Draggable**

First of all: To make an element draggable, set the draggable attribute to true:

<img draggable="true">

## **What to Drag - ondragstart and setData()**

Then, specify what should happen when the element is dragged.

In the example above, the ondragstart attribute calls a function, drag(event), that specifies what data to be dragged.

The dataTransfer.setData() method sets the data type and the value of the dragged data:

function drag(ev) {  
  ev.dataTransfer.setData("text", ev.target.id);  
}

In this case, the data type is "text" and the value is the id of the draggable element ("drag1").

## **Where to Drop - ondragover**

The ondragover event specifies where the dragged data can be dropped.

By default, data/elements cannot be dropped in other elements. To allow a drop, we must prevent the default handling of the element.

This is done by calling the event.preventDefault() method for the ondragover event:

*event*.preventDefault()

## **Do the Drop - ondrop**

When the dragged data is dropped, a drop event occurs.

In the example above, the ondrop attribute calls a function, drop(event):

function drop(ev) {  
  ev.preventDefault();  
  var data = ev.dataTransfer.getData("text");  
  ev.target.appendChild(document.getElementById(data));  
}

Code explained:

* Call preventDefault() to prevent the browser default handling of the data (default is open as link on drop)
* Get the dragged data with the dataTransfer.getData() method. This method will return any data that was set to the same type in the setData() method
* The dragged data is the id of the dragged element ("drag1")
* Append the dragged element into the drop element

## **More Examples**

### Example

How to drag (and drop) an image back and forth between two <div> elements:

# **HTML**Web Storage API

HTML web storage; better than cookies.

## **What is HTML Web Storage?**

With web storage, web applications can store data locally within the user's browser.

Before HTML5, application data had to be stored in cookies, included in every server request. Web storage is more secure, and large amounts of data can be stored locally, without affecting website performance.

Unlike cookies, the storage limit is far larger (at least 5MB) and information is never transferred to the server.

Web storage is per origin (per domain and protocol). All pages, from one origin, can store and access the same data.

## **Browser Support**

The numbers in the table specify the first browser version that fully supports Web Storage.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| API |  |  |  |  |  |
| Web Storage | 4.0 | 8.0 | 3.5 | 4.0 | 11.5 |

## **HTML Web Storage Objects**

HTML web storage provides two objects for storing data on the client:

* window.localStorage - stores data with no expiration date
* window.sessionStorage - stores data for one session (data is lost when the browser tab is closed)

Before using web storage, check browser support for localStorage and sessionStorage:

if (typeof(Storage) !== "undefined") {  
  // *Code for localStorage/sessionStorage.*  
} else {  
  // Sorry! No Web Storage support..  
}

## **The localStorage Object**

The localStorage object stores the data with no expiration date. The data will not be deleted when the browser is closed, and will be available the next day, week, or year.

### Example

// Store  
localStorage.setItem("lastname", "Smith");  
  
// Retrieve  
document.getElementById("result").innerHTML = localStorage.getItem("lastname");

Example explained:

* Create a localStorage name/value pair with name="lastname" and value="Smith"
* Retrieve the value of "lastname" and insert it into the element with id="result"

The example above could also be written like this:

// Store  
localStorage.lastname = "Smith";  
// Retrieve  
document.getElementById("result").innerHTML = localStorage.lastname;

The syntax for removing the "lastname" localStorage item is as follows:

localStorage.removeItem("lastname");

**Note:** Name/value pairs are always stored as strings. Remember to convert them to another format when needed!

The following example counts the number of times a user has clicked a button. In this code the value string is converted to a number to be able to increase the counter:

### Example

if (localStorage.clickcount) {  
  localStorage.clickcount = Number(localStorage.clickcount) + 1;  
} else {  
  localStorage.clickcount = 1;  
}  
document.getElementById("result").innerHTML = "You have clicked the button " +  
localStorage.clickcount + " time(s).";

## **The sessionStorage Object**

The sessionStorage object is equal to the localStorage object, **except** that it stores the data for only one session. The data is deleted when the user closes the specific browser tab.

The following example counts the number of times a user has clicked a button, in the current session:

### Example

if (sessionStorage.clickcount) {  
  sessionStorage.clickcount = Number(sessionStorage.clickcount) + 1;  
} else {  
  sessionStorage.clickcount = 1;  
}  
document.getElementById("result").innerHTML = "You have clicked the button " +  
sessionStorage.clickcount + " time(s) in this session.";

# **HTML**Web Workers API

A web worker is a JavaScript running in the background, without affecting the performance of the page.

## **What is a Web Worker?**

When executing scripts in an HTML page, the page becomes unresponsive until the script is finished.

A web worker is a JavaScript that runs in the background, independently of other scripts, without affecting the performance of the page. You can continue to do whatever you want: clicking, selecting things, etc., while the web worker runs in the background.

## **Browser Support**

The numbers in the table specify the first browser version that fully support Web Workers.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| API |  |  |  |  |  |
| Web Workers | 4.0 | 10.0 | 3.5 | 4.0 | 11.5 |

## **HTML Web Workers Example**

The example below creates a simple web worker that count numbers in the background:

### Example

Count numbers:

Start Worker Stop Worker

## **Check Web Worker Support**

Before creating a web worker, check whether the user's browser supports it:

if (typeof(Worker) !== "undefined") {  
  // Yes! Web worker support!  
  // *Some code.....*  
} else {  
  // Sorry! No Web Worker support..  
}

## **Create a Web Worker File**

Now, let's create our web worker in an external JavaScript.

Here, we create a script that counts. The script is stored in the "demo\_workers.js" file:

var i = 0;  
  
function timedCount() {  
  i = i + 1;  
  postMessage(i);  
  setTimeout("timedCount()",500);  
}  
  
timedCount();

The important part of the code above is the postMessage() method - which is used to post a message back to the HTML page.

**Note:** Normally web workers are not used for such simple scripts, but for more CPU intensive tasks.

## **Create a Web Worker Object**

Now that we have the web worker file, we need to call it from an HTML page.

The following lines checks if the worker already exists, if not - it creates a new web worker object and runs the code in "demo\_workers.js":

if (typeof(w) == "undefined") {  
  w = new Worker("demo\_workers.js");  
}

Then we can send and receive messages from the web worker.

Add an "onmessage" event listener to the web worker.

w.onmessage = function(event){  
  document.getElementById("result").innerHTML = event.data;  
};

When the web worker posts a message, the code within the event listener is executed. The data from the web worker is stored in event.data.

## **Terminate a Web Worker**

When a web worker object is created, it will continue to listen for messages (even after the external script is finished) until it is terminated.

To terminate a web worker, and free browser/computer resources, use the terminate() method:

w.terminate();

## **Reuse the Web Worker**

If you set the worker variable to undefined, after it has been terminated, you can reuse the code:

w = undefined;

## **Full Web Worker Example Code**

We have already seen the Worker code in the .js file. Below is the code for the HTML page:

### Example

<!DOCTYPE html>  
<html>  
<body>  
  
<p>Count numbers: <output id="result"></output></p>  
<button onclick="startWorker()">Start Worker</button>  
<button onclick="stopWorker()">Stop Worker</button>  
  
<script>  
var w;  
  
function startWorker() {  
  if (typeof(Worker) !== "undefined") {  
    if (typeof(w) == "undefined") {  
      w = new Worker("demo\_workers.js");  
    }  
    w.onmessage = function(event) {  
      document.getElementById("result").innerHTML = event.data;  
    };  
  } else {  
    document.getElementById("result").innerHTML = "Sorry! No Web Worker support.";  
  }  
}  
  
function stopWorker() {  
  w.terminate();  
  w = undefined;  
}  
</script>  
  
</body>  
</html>

## **Web Workers and the DOM**

Since web workers are in external files, they do not have access to the following JavaScript objects:

* The window object
* The document object
* The parent object

# **HTML**SSE API

Server-Sent Events (SSE) allow a web page to get updates from a server.

## **Server-Sent Events - One Way Messaging**

A server-sent event is when a web page automatically gets updates from a server.

This was also possible before, but the web page would have to ask if any updates were available. With server-sent events, the updates come automatically.

Examples: Facebook/Twitter updates, stock price updates, news feeds, sport results, etc.

## **Browser Support**

The numbers in the table specify the first browser version that fully support server-sent events.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| API |  |  |  |  |  |
| SSE | 6.0 | 79.0 | 6.0 | 5.0 | 11.5 |

## **Receive Server-Sent Event Notifications**

The EventSource object is used to receive server-sent event notifications:

### Example

var source = new EventSource("demo\_sse.php");  
source.onmessage = function(event) {  
  document.getElementById("result").innerHTML += event.data + "<br>";  
};

Example explained:

* Create a new EventSource object, and specify the URL of the page sending the updates (in this example "demo\_sse.php")
* Each time an update is received, the onmessage event occurs
* When an onmessage event occurs, put the received data into the element with id="result"

## **Check Server-Sent Events Support**

In the tryit example above there were some extra lines of code to check browser support for server-sent events:

if(typeof(EventSource) !== "undefined") {  
  // Yes! Server-sent events support!  
  // *Some code.....*  
} else {  
  // Sorry! No server-sent events support..  
}

## **Server-Side Code Example**

For the example above to work, you need a server capable of sending data updates (like PHP or ASP).

The server-side event stream syntax is simple. Set the "Content-Type" header to "text/event-stream". Now you can start sending event streams.

Code in PHP (demo\_sse.php):

<?php  
header('Content-Type: text/event-stream');  
header('Cache-Control: no-cache');  
  
$time = date('r');  
echo "data: The server time is: {$time}\n\n";  
flush();  
?>

Code in ASP (VB) (demo\_sse.asp):

<%  
Response.ContentType = "text/event-stream"  
Response.Expires = -1  
Response.Write("data: The server time is: " & now())  
Response.Flush()  
%>

Code explained:

* Set the "Content-Type" header to "text/event-stream"
* Specify that the page should not cache
* Output the data to send (**Always** start with "data: ")
* Flush the output data back to the web page

## **The EventSource Object**

In the examples above we used the onmessage event to get messages. But other events are also available:

|  |  |
| --- | --- |
| **Events** | **Description** |
| onopen | When a connection to the server is opened |
| onmessage | When a message is received |
| onerror | When an error occurs |

# **, 100%, 64%, 0.6)** HTML Multimedia

Multimedia on the web is sound, music, videos, movies, and animations.

## **What is Multimedia?**

Multimedia comes in many different formats. It can be almost anything you can hear or see, like images, music, sound, videos, records, films, animations, and more.

Web pages often contain multimedia elements of different types and formats.

## **Browser Support**

The first web browsers had support for text only, limited to a single font in a single color.

Later came browsers with support for colors, fonts, images, and multimedia!

## **Multimedia Formats**

Multimedia elements (like audio or video) are stored in media files.

The most common way to discover the type of a file, is to look at the file extension.

Multimedia files have formats and different extensions like: .wav, .mp3, .mp4, .mpg, .wmv, and .avi.

## **Common Video Formats**

|  |  |
| --- | --- |
| Videoformats | There are many video formats out there.  The MP4, WebM, and Ogg formats are supported by HTML.  The MP4 format is recommended by YouTube. |

|  |  |  |
| --- | --- | --- |
| **Format** | **File** | **Description** |
| MPEG | .mpg .mpeg | MPEG. Developed by the Moving Pictures Expert Group. The first popular video format on the web. Not supported anymore in HTML. |
| AVI | .avi | AVI (Audio Video Interleave). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers. |
| WMV | .wmv | WMV (Windows Media Video). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers. |
| QuickTime | .mov | QuickTime. Developed by Apple. Commonly used in video cameras and TV hardware. Plays well on Apple computers, but not in web browsers. |
| RealVideo | .rm .ram | RealVideo. Developed by Real Media to allow video streaming with low bandwidths. Does not play in web browsers. |
| Flash | .swf .flv | Flash. Developed by Macromedia. Often requires an extra component (plug-in) to play in web browsers. |
| Ogg | .ogg | Theora Ogg. Developed by the Xiph.Org Foundation. Supported by HTML. |
| WebM | .webm | WebM. Developed by Mozilla, Opera, Adobe, and Google. Supported by HTML. |
| MPEG-4 or MP4 | .mp4 | MP4. Developed by the Moving Pictures Expert Group. Commonly used in video cameras and TV hardware. Supported by all browsers and  recommended by YouTube. |

**Note:** Only MP4, WebM, and Ogg video are supported by the HTML standard.

## **Common Audio Formats**

MP3 is the best format for compressed recorded music. The term MP3 has become synonymous with digital music.

If your website is about recorded music, MP3 is the choice.

|  |  |  |
| --- | --- | --- |
| **Format** | **File** | **Description** |
| MIDI | .mid .midi | MIDI (Musical Instrument Digital Interface). Main format for all electronic music devices like synthesizers and PC sound cards. MIDI files do not contain sound, but digital notes that can be played by electronics. Plays well on all computers and music hardware, but not in web browsers. |
| RealAudio | .rm .ram | RealAudio. Developed by Real Media to allow streaming of audio with low bandwidths. Does not play in web browsers. |
| WMA | .wma | WMA (Windows Media Audio). Developed by Microsoft. Plays well on Windows computers, but not in web browsers. |
| AAC | .aac | AAC (Advanced Audio Coding). Developed by Apple as the default format for iTunes. Plays well on Apple computers, but not in web browsers. |
| WAV | .wav | WAV. Developed by IBM and Microsoft. Plays well on Windows, Macintosh, and Linux operating systems. Supported by HTML. |
| Ogg | .ogg | Ogg. Developed by the Xiph.Org Foundation. Supported by HTML. |
| MP3 | .mp3 | MP3 files are actually the sound part of MPEG files. MP3 is the most popular format for music players. Combines good compression (small files) with high quality. Supported by all browsers. |
| MP4 | .mp4 | MP4 is a video format, but can also be used for audio. Supported by all browsers. |

**Note:** Only MP3, WAV, and Ogg audio are supported by the HTML standard.

# HTML Video

The HTML <video> element is used to show a video on a web page.

### Example

Courtesy of [Big Buck Bunny](https://www.bigbuckbunny.org/):

<!DOCTYPE html>

<html>

<body>

<video width="400" controls>

<source src="mov\_bbb.mp4" type="video/mp4">

<source src="mov\_bbb.ogg" type="video/ogg">

Your browser does not support HTML video.

</video>

<p>

Video courtesy of

<a href="https://www.bigbuckbunny.org/" target="\_blank">Big Buck Bunny</a>.

</p>

</body>

</html>

**OUTPUT**

Video courtesy of [Big Buck Bunny](https://www.bigbuckbunny.org/).

## **The HTML <video> Element**

To show a video in HTML, use the <video> element:

<!DOCTYPE html>

<html>

<body>

<video width="320" height="240" controls>

<source src="movie.mp4" type="video/mp4">

<source src="movie.ogg" type="video/ogg">

Your browser does not support the video tag.

</video>

</body>

</html>

## **How it Works**

The controls attribute adds video controls, like play, pause, and volume.

It is a good idea to always include width and height attributes. If height and width are not set, the page might flicker while the video loads.

The <source> element allows you to specify alternative video files which the browser may choose from. The browser will use the first recognized format.

The text between the <video> and </video> tags will only be displayed in browsers that do not support the <video> element.

## **HTML <video> Autoplay**

To start a video automatically, use the autoplay attribute:

### Example

<!DOCTYPE html>

<html>

<body>

<video width="320" height="240" autoplay>

<source src="movie.mp4" type="video/mp4">

<source src="movie.ogg" type="video/ogg">

Your browser does not support the video tag.

</video>

</body>

</html>

**Note:** Chromium browsers do not allow autoplay in most cases. However, muted autoplay is always allowed.

Add muted after autoplay to let your video start playing automatically (but muted):

### Example

<!DOCTYPE html>

<html>

<body>

<video width="320" height="240" autoplay muted>

<source src="movie.mp4" type="video/mp4">

<source src="movie.ogg" type="video/ogg">

Your browser does not support the video tag.

</video>

</body>

</html>

## **Browser Support**

The numbers in the table specify the first browser version that fully supports the <video> element.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Element |  |  |  |  |  |
| <video> | 4.0 | 9.0 | 3.5 | 4.0 | 10.5 |

## **HTML Video Formats**

There are three supported video formats: MP4, WebM, and Ogg. The browser support for the different formats is:

|  |  |  |  |
| --- | --- | --- | --- |
| **Browser** | **MP4** | **WebM** | **Ogg** |
| Edge | YES | YES | YES |
| Chrome | YES | YES | YES |
| Firefox | YES | YES | YES |
| Safari | YES | YES | NO |
| Opera | YES | YES | YES |

## **HTML Video - Media Types**

|  |  |
| --- | --- |
| **File Format** | **Media Type** |
| MP4 | video/mp4 |
| WebM | video/webm |
| Ogg | video/ogg |

## **HTML Video - Methods, Properties, and Events**

The HTML DOM defines methods, properties, and events for the <video> element.

This allows you to load, play, and pause videos, as well as setting duration and volume.

There are also DOM events that can notify you when a video begins to play, is paused, etc.

### Example: Using JavaScript

For a full DOM reference, go to our [HTML Audio/Video DOM Reference](https://www.w3schools.com/tags/ref_av_dom.asp).

## **HTML Video Tags**

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<video>](https://www.w3schools.com/tags/tag_video.asp) | Defines a video or movie |
| [<source>](https://www.w3schools.com/tags/tag_source.asp) | Defines multiple media resources for media elements, such as <video> and <audio> |
| [<track>](https://www.w3schools.com/tags/tag_track.asp) | Defines text tracks in media players |

# HTML Audio

The HTML <audio> element is used to play an audio file on a web page.

## **The HTML <audio> Element**

To play an audio file in HTML, use the <audio> element:

### Example

# **<!DOCTYPE html>**

# **<html>**

# **<body>**

# **<audio controls>**

# **<source src="horse.ogg" type="audio/ogg">**

# **<source src="horse.mp3" type="audio/mpeg">**

# **Your browser does not support the audio element.**

# **</audio>**

# **</body>**

# **</html>**

## **HTML Audio - How It Works**

The controls attribute adds audio controls, like play, pause, and volume.

The <source> element allows you to specify alternative audio files which the browser may choose from. The browser will use the first recognized format.

The text between the <audio> and </audio> tags will only be displayed in browsers that do not support the <audio> element.

## **HTML <audio> Autoplay**

To start an audio file automatically, use the autoplay attribute:

### Example

## **HTML Audio - How It Works**

The controls attribute adds audio controls, like play, pause, and volume.

The <source> element allows you to specify alternative audio files which the browser may choose from. The browser will use the first recognized format.

The text between the <audio> and </audio> tags will only be displayed in browsers that do not support the <audio> element.

## **HTML <audio> Autoplay**

To start an audio file automatically, use the autoplay attribute:

### Example

<!DOCTYPE html>

<html>

<body>

<audio controls autoplay>

<source src="horse.ogg" type="audio/ogg">

<source src="horse.mp3" type="audio/mpeg">

Your browser does not support the audio element.

</audio>

</body>

</html>

## **Browser Support**

The numbers in the table specify the first browser version that fully supports the <audio> element.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Element |  |  |  |  |  |
| <audio> | 4.0 | 9.0 | 3.5 | 4.0 | 10.5 |

## **HTML Audio Formats**

There are three supported audio formats: MP3, WAV, and OGG. The browser support for the different formats is:

|  |  |  |  |
| --- | --- | --- | --- |
| **Browser** | **MP3** | **WAV** | **OGG** |
| Edge/IE | YES | YES\* | YES\* |
| Chrome | YES | YES | YES |
| Firefox | YES | YES | YES |
| Safari | YES | YES | NO |
| Opera | YES | YES | YES |

\*From Edge 79

## **HTML Audio - Media Types**

|  |  |
| --- | --- |
| **File Format** | **Media Type** |
| MP3 | audio/mpeg |
| OGG | audio/ogg |

## HTML Audio - Methods, Properties, and Events

The HTML DOM defines methods, properties, and events for the <audio> element.

This allows you to load, play, and pause audios, as well as set duration and volume.

There are also DOM events that can notify you when an audio begins to play, is paused, etc.

## HTML Audio Tags

|  |  |
| --- | --- |
| Tag | Description |
| [<audio>](https://www.w3schools.com/tags/tag_audio.asp) | Defines sound content |
| [<source>](https://www.w3schools.com/tags/tag_source.asp) | Defines multiple media resources for media elements, such as <video> and <audio> |

# **HTML Plug-ins**

Plug-ins are computer programs that extend the standard functionality of the browser.

## **Plug-ins**

Plug-ins were designed to be used for many different purposes:

* To run Java applets
* To run Microsoft ActiveX controls
* To display Flash movies
* To display maps
* To scan for viruses
* To verify a bank id

**Warning !**

Most browsers no longer support Java Applets and Plug-ins.

ActiveX controls are no longer supported in any browsers.

The support for Shockwave Flash has also been turned off in modern browsers.

## **The <object> Element**

The <object> element is supported by all browsers.

The <object> element defines an embedded object within an HTML document.

It was designed to embed plug-ins (like Java applets, PDF readers, and Flash Players) in web pages, but can also be used to include HTML in HTML:

### Example

<!DOCTYPE html>

<html>

<body>

<object width="100%" height="500px" data="snippet.html"></object>

</body>

</html>

**OUTPUT**

|  |  |  |
| --- | --- | --- |
| Alfreds Futterkiste | Berlin | Germany |
| Berglunds snabbköp | Luleå | Sweden |
| Centro comercial Moctezuma | México D.F. | Mexico |
| Ernst Handel | Graz | Austria |
| FISSA Fabrica Inter. Salchichas S.A. | Madrid | Spain |
| Galería del gastrónomo | Barcelona | Spain |
| Island Trading | Cowes | UK |
| Königlich Essen | Brandenburg | Germany |
| Laughing Bacchus Wine Cellars | Vancouver | Canada |
| Magazzini Alimentari Riuniti | Bergamo | Italy |
| North/South | London | UK |
| Paris spécialités | Paris | France |
| Rattlesnake Canyon Grocery | Albuquerque | USA |
| Simons bistro | København | Denmark |

**EXAMPLE**

<!DOCTYPE html>

<html>

<body>

<object data="audi.jpeg"></object>

</body>

</html>

## **The <embed> Element**

The <embed> element is supported in all major browsers.

The <embed> element also defines an embedded object within an HTML document.

Web browsers have supported the <embed> element for a long time. However, it has not been a part of the HTML specification before HTML5.

### Example

<!DOCTYPE html>

<html>

<body>

<embed src="audi.jpeg">

</body>

</html>

Note that the <embed> element does not have a closing tag. It can not contain alternative text.

The <embed> element can also be used to include HTML in HTML:

**Example**

### <!DOCTYPE html>

### <html>

### <body>

### <embed width="100%" height="500px" src="snippet.html">

### </body>

### </html>

### OUTPUT

|  |  |  |
| --- | --- | --- |
| Alfreds Futterkiste | Berlin | Germany |
| Berglunds snabbköp | Luleå | Sweden |
| Centro comercial Moctezuma | México D.F. | Mexico |
| Ernst Handel | Graz | Austria |
| FISSA Fabrica Inter. Salchichas S.A. | Madrid | Spain |
| Galería del gastrónomo | Barcelona | Spain |
| Island Trading | Cowes | UK |
| Königlich Essen | Brandenburg | Germany |
| Laughing Bacchus Wine Cellars | Vancouver | Canada |
| Magazzini Alimentari Riuniti | Bergamo | Italy |
| North/South | London | UK |
| Paris spécialités | Paris | France |
| Rattlesnake Canyon Grocery | Albuquerque | USA |
| Simons bistro | København | Denmark |
| The Big Cheese | Portland | USA |
| Vaffeljernet | Århus | Denmark |
| Wolski Zajazd | Warszawa | Poland |

# HTML YouTube Videos

The easiest way to play videos in HTML, is to use YouTube.

## **Struggling with Video Formats?**

Converting videos to different formats can be difficult and time-consuming.

An easier solution is to let YouTube play the videos in your web page.

## **YouTube Video Id**

YouTube will display an id (like tgbNymZ7vqY), when you save (or play) a video.

You can use this id, and refer to your video in the HTML code.

## **Playing a YouTube Video in HTML**

To play your video on a web page, do the following:

* Upload the video to YouTube
* Take a note of the video id
* Define an <iframe> element in your web page
* Let the src attribute point to the video URL
* Use the width and height attributes to specify the dimension of the player
* Add any other parameters to the URL (see below)

### Example

### <!DOCTYPE html>

### <html>

### <body>

### <iframe width="420" height="345" src="https://www.youtube.com/embed/tgbNymZ7vqY">

### </iframe>

### </body>

## **YouTube Autoplay + Mute**

You can let your video start playing automatically when a user visits the page, by adding autoplay=1 to the YouTube URL. However, automatically starting a video is annoying for your visitors!

**Note:** Chromium browsers do not allow autoplay in most cases. However, muted autoplay is always allowed.

Add mute=1 after autoplay=1 to let your video start playing automatically (but muted).

### YouTube - Autoplay + Muted

## **YouTube Autoplay + Mute**

You can let your video start playing automatically when a user visits the page, by adding autoplay=1 to the YouTube URL. However, automatically starting a video is annoying for your visitors!

**Note:** Chromium browsers do not allow autoplay in most cases. However, muted autoplay is always allowed.

Add mute=1 after autoplay=1 to let your video start playing automatically (but muted).

### YouTube - Autoplay + Muted

## **YouTube Playlist**

A comma separated list of videos to play (in addition to the original URL).

## **YouTube Loop**

Add loop=1 to let your video loop forever.

Value 0 (default): The video will play only once.

Value 1: The video will loop (forever).

### YouTube - Loop

## **YouTube Controls**

Add controls=0 to not display controls in the video player.

Value 0: Player controls does not display.

Value 1 (default): Player controls display.

### YouTube - Controls

<iframe width="420" height="315"  
src="https://www.youtube.com/embed/tgbNymZ7vqY?controls=0">  
</iframe>

**OUTPUT**

<!DOCTYPE html>

<html>

<body>

<iframe width="420" height="345" src="https://www.youtube.com/embed/tgbNymZ7vqY?controls=0">

</iframe>

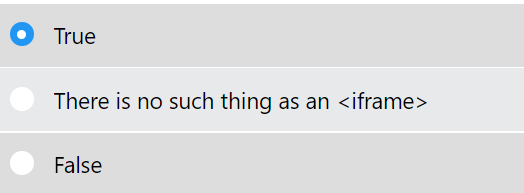
</body>

</html>

HTML Quiz

Question 1:

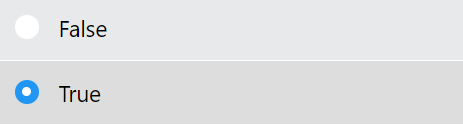
An <iframe> is used to display a web page within a web page.



HTML Quiz

Question 2:

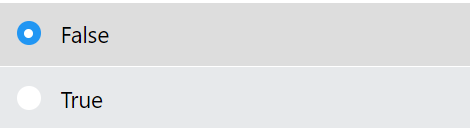
HTML comments start with <!-- and end with -->



HTML Quiz

Question 3:

Block elements are normally displayed without starting a new line.



# **HTML Quiz**

### Question 4:

Which HTML element defines the title of a document?



# **HTML Quiz**

### Question 5:

Which HTML attribute specifies an alternate text for an image, if the image cannot be displayed?



HTML Quiz

Question 6:

Which doctype is correct for HTML5?



# **HTML Quiz**

### Question 7:

Which HTML element is used to specify a footer for a document or section?



# **HTML Quiz**

### Question 8:

In HTML, you can embed SVG elements directly into an HTML page.



# **HTML Quiz**

### Question 9:

What is the correct HTML element for playing video files?



# **HTML Quiz**

### Question 10:

What is the correct HTML element for playing audio files?



**hs9, 100%, 64%, 0. %, 64%, 1)**

**r55, 99, 71, 0.5)**

**100%, 64%, 0.5)**