

# HTML Tutorial

With HTML you can create your own Web site.

This tutorial teaches you everything about HTML.

HTML is easy to learn - You will enjoy it.

## Examples in Every Chapter

This HTML tutorial contains hundreds of HTML examples.

With our online HTML editor, you can edit the HTML, and click on a button to view the result.

### 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 a **markup** language for **describing** web documents (web pages).

- HTML stands for **H**yper **T**ext **M**arkup **L**anguage
- A markup language is a set of **markup tags**
- HTML documents are described by **HTML tags**
- Each HTML tag **describes** different document content

# HTML Example

## A small HTML document:

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

<body>

  <h1>My First Heading</h1>

  <p>My first paragraph.</p>

</body>

</html>
```

## Example Explained

- The **DOCTYPE** declaration defines the document type to be HTML
- The text between **<html>** and **</html>** describes an HTML document
- The text between **<head>** and **</head>** provides information about the document
- The text between **<title>** and **</title>** provides a title for the document
- The text between **<body>** and **</body>** describes the visible page content
- The text between **<h1>** and **</h1>** describes a heading
- The text between **<p>** and **</p>** describes paragraph

Using this description, a web browser can display a document with a heading and a paragraph.

---

## HTML Tags

HTML tags are **keywords** (tag names) surrounded by **angle brackets**:

**<tagname>**content**</tagname>**

- HTML tags normally come **in pairs** like **<p>** and **</p>**
- The first tag in a pair is the **start tag**, the second tag is the **end tag**
- The end tag is written like the start tag, but with a **slash** before the tag name



The start tag is often called the **opening tag**. The end tag is often called the **closing tag**.

# Web Browsers

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

The 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>
```



Only the <body> area (the white area) is displayed by the browser.

## The <!DOCTYPE> Declaration

The <!DOCTYPE> declaration helps the browser to display a web page correctly.

There are different document types on the web.

To display a document correctly, the browser must know both type and version.

The doctype declaration is not case sensitive. All cases are acceptable:

<!DOCTYPE html>

<!DOCTYPE HTML>

<!doctype html>

<!Doctype Html>

## HTML Versions

Since the early days of the web, there have been many versions of HTML:

Version	Year
HTML	1991
HTML 2.0	1995
HTML 3.2	1997
HTML 4.01	1999
XHTML	2000
HTML5	2012

## Write HTML Using Notepad or TextEdit

HTML can be edited by using a professional HTML editor like:

- Adobe Dreamweaver
- Microsoft Expression Web
- CoffeeCup HTML Editor

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

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

Follow the 4 steps below to create your first web page with Notepad.

---

## Step 1: Open Notepad

To open Notepad in Windows 7 or earlier:

Click **Start** (bottom left on your screen). Click **All Programs**. Click **Accessories**. Click **Notepad**.

To open Notepad in Windows 8 or later:

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

---

## Step 2: Write Some HTML

Write or copy some HTML into Notepad.

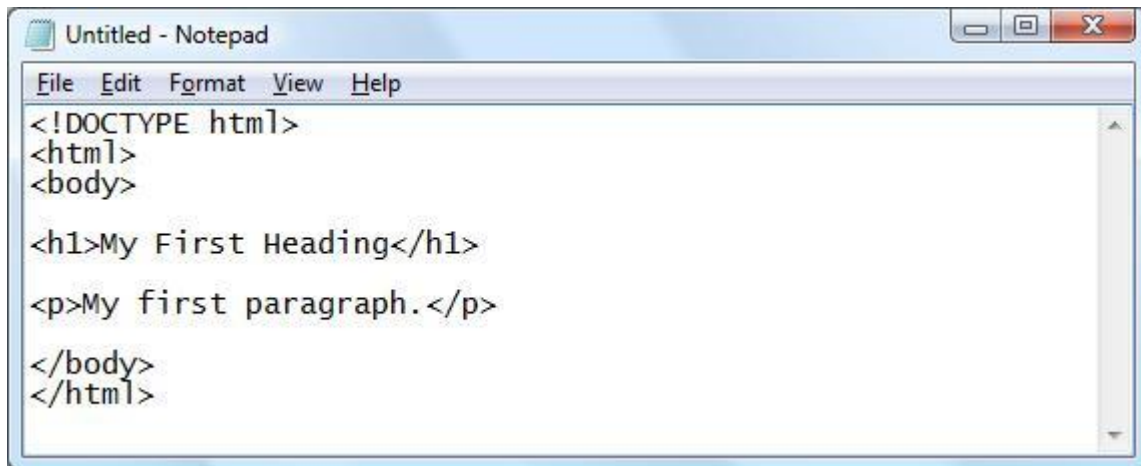
### Example

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

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```



---

## Step 3: Save the HTML Page

Save the file on your computer.

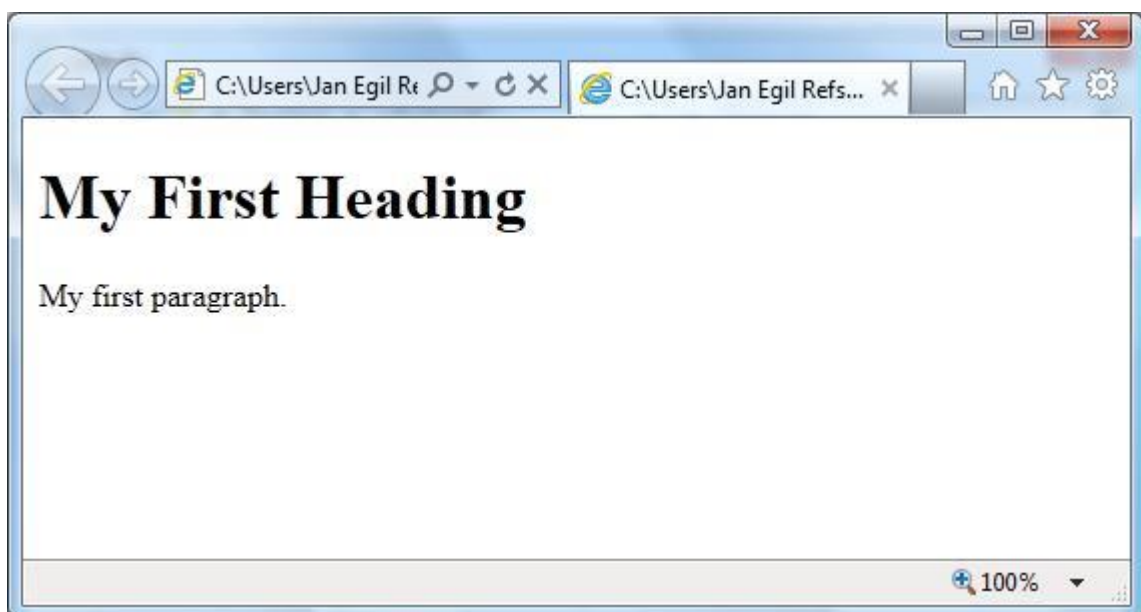
Select **File > Save as** in the Notepad menu.

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

---

## Step 4: View HTML Page in Your Browser

Double-click your saved HTML file, and the result will look much like this:



# HTML Basic Examples

---

Don't worry if these examples use tags you have not learned.

You will learn them in the next chapters.

---

## HTML Documents

All HTML documents must start with a 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>
```

---

## HTML Headings

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

### Example

```
<h1>This is a heading</h1>
<h2>This is a heading</h2>
<h3>This is a heading</h3>
```

---

## HTML Paragraphs

HTML paragraphs are defined with the **<p>** tag:

## Example

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

---

## HTML Links

HTML links are defined with the **<a>** tag:

### Example

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

The link address is specified in the **href attribute**.

Attributes are used to provide additional information about HTML elements.

---

## HTML Images

HTML images are defined with the **<img>** tag.

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

### Example

```

```



You will learn more about attributes in a later chapter.

## HTML Elements

HTML **documents** are made up by HTML **elements**.

---

## HTML Elements

HTML elements are written with a **start** tag, with an **end** tag, with the **content** in between:


```
<tagname>content</tagname>
```

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



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

Start tag	Element content	End tag
<code>&lt;h1&gt;</code>	My First Heading	<code>&lt;/h1&gt;</code>
<code>&lt;p&gt;</code>	My first paragraph.	<code>&lt;/p&gt;</code>
<code>&lt;br&gt;</code>		

 Some HTML elements do not have an end tag.

## Nested HTML Elements

HTML elements can be nested (elements can contain elements).

All HTML documents consist of nested HTML elements.

This example contains 4 HTML elements:

### Example

```
<!DOCTYPE html>
<html>

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

</html>
```

## HTML Example Explained

The `<html>` element defines the **whole document**.

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

The element **content** is another HTML element (the `<body>` element).

```
<html>
```

```
<body>
```

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

</html>
```

The **<body>** element defines the **document body**.

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

The element **content** is two other HTML elements (`<h1>` and `<p>`).

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

The **<h1>** element defines a **heading**.

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

The element **content** is: My First Heading.

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

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

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

The element **content** is: My first paragraph.

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

---

## Don't Forget the End Tag

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

### Example

```
<html>

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

</html>
```

The example above works in all browsers, because the closing tag is considered optional.

Never rely on this. It might produce unexpected results and/or errors if you forget the end tag.

---

## Empty HTML Elements

HTML elements with no content are called empty elements.

`<br>` is an empty element without a closing tag (the `<br>` tag defines a line break).

Empty element can be "closed" in the opening tag like this: `<br />`.

HTML5 does not require empty elements to be closed. But if you need stricter validation, and make your document readable by XML parsers, please close all HTML elements.

---

## HTML Tip: Use Lowercase Tags

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

The HTML5 standard does not require lowercase tags, but W3C **recommends** lowercase in HTML4, and **demands** lowercase for stricter document types like XHTML.



At W3Schools we always use lowercase tags.

## HTML Attributes

Attributes provide additional information about HTML elements.

---

### HTML Attributes

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

# The lang Attribute

The document language can be declared in the **<html>** tag.

The language is declared in the **lang** attribute.

Declaring a language is important for accessibility applications (screen readers) and search engines:

## Example

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

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```

The first two letters specify the language (en). If there is a dialect, use two more letters (US).

---

# The title Attribute

HTML paragraphs are defined with the **<p>** tag.

In this example, the **<p>** element has a **title** attribute. The value of the attribute is **"About W3Schools"**:

## Example

```
<p title="About W3Schools">
W3Schools is a web developer's site.
It provides tutorials and references covering
many aspects of web programming,
including HTML, CSS, JavaScript, XML, SQL, PHP, ASP, etc.
</p>
```

When you move the mouse over the element, the title will be displayed as a tooltip.

---

# The href Attribute

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

## Example

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

You will learn more about links and the <a> tag later in this tutorial.

---

## Size Attributes

HTML images are defined with the <img> tag.

The filename of the source (**src**), and the size of the image (**width** and **height**) are all provided as **attributes**:

### Example

```

```

The image size is specified in pixels: width="104" means 104 screen pixels wide.

You will learn more about images and the <img> tag later in this tutorial.

---

## The alt Attribute

The **alt** attribute specifies an alternative text to be used, when an HTML element cannot be displayed.

The value of the attribute can be read by "screen readers". This way, someone "listening" to the webpage, i.e. a blind person, can "hear" the element.

### Example

```

```

## We Suggest: Always Use Lowercase Attributes

The HTML5 standard does not require lower case attribute names.

The title attribute can be written with upper or lower case like **Title** and/or **TITLE**.

W3C **recommends** lowercase in HTML4, and **demands** lowercase for stricter document types like XHTML.



Lower case is the most common. Lower case is easier to type.  
At W3Schools we always use lower case attribute names.

---

## We Suggest: Always Quote Attribute Values

The HTML5 standard does not require quotes around attribute values.

The **href** attribute, demonstrated above, can be written as:

### Example

```
<a href=http://www.w3schools.com>
```

## Single or Double Quotes?

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

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

### Example

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

Or vice versa:

### Example

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

---

## Chapter Summary

- All HTML elements can have **attributes**
- The HTML **title** attribute provides additional "tool-tip" information
- The HTML **href** attribute provides address information for links
- The HTML **width** and **height** attributes provide size information for images
- The HTML **alt** attribute provides text for screen readers
- At W3Schools we always use **lowercase** HTML attribute names
- At W3Schools we always **quote** attributes with double quotes

## HTML Attributes

Below is an alphabetical list of some attributes often used in HTML:

Attribute	Description
alt	Specifies an alternative text for an image

disabled	Specifies that an input element should be disabled
href	Specifies the URL (web address) for a link
id	Specifies a unique id for an element
src	Specifies the URL (web address) for an image
style	Specifies an inline CSS style for an element
title	Specifies extra information about an element (displayed as a tool tip)
value	Specifies the value (text content) for an input element.

## HTML Headings

### HTML Headings

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 a heading</h1>
<h2>This is a heading</h2>
<h3>This is a heading</h3>
```

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

---

## Headings Are Important

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

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

Users skim your pages by its headings. It is important to use headings to show the document structure.

h1 headings should be main headings, followed by h2 headings, then the less important h3, and so on.

---

## HTML Horizontal Rules

The **<hr>** tag creates a horizontal line in an HTML page.

The hr element can be used to separate content:

### Example

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

## The HTML <head> Element

The HTML **<head>** element has nothing to do with HTML headings.

The HTML <head> element contains **meta data**. Meta data are not displayed.

The HTML <head> element is placed between the <html> tag and the <body> tag:

### Example

```
<!DOCTYPE html>
<html>

<head>
  <title>My First HTML</title>
  <meta charset="UTF-8">
</head>

<body>
  .
  .
  .
```

## HTML Paragraphs

HTML documents are divided into paragraphs.

---

## HTML Paragraphs

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



## Example

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



Browsers automatically add an empty line before and after a paragraph.

## HTML Display

You cannot be sure how HTML will be displayed.

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

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

The browser will remove extra spaces and extra lines when the page is displayed.

Any number of spaces, and any number of new lines, count as **only one space**.

## Example

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

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

## Don't Forget the End Tag

Most browsers will display HTML correctly even if you forget the end tag:

## Example

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

The example above will work in most browsers, but don't rely on it.

Forgetting the end tag can produce unexpected results or errors.



Stricter versions of HTML, like XHTML, do not allow you to skip the 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 para<br>graph with line breaks</p>
```

The **<br>** element is an empty HTML element. It has no end tag.

---

## The Poem Problem

### Example

```
<p>This poem will display as one line:</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>
```

## HTML Styles

### HTML Styling

Every HTML element has a **default style** (background color is white, text color is black, text-size is 12px ...)

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

This example changes the default background color from white to lightgrey:

### Example

```
<body style="background-color:lightgrey">

<h1>This is a heading</h1>

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

</body>
```

## The HTML Style Attribute

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

`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.

---

## HTML Text Color

The **color** property defines the text color to be used for an HTML element:

### Example

```
<!DOCTYPE html>
<html>

<body>
  <h1 style="color:blue">This is a heading</h1>
  <p style="color:red">This is a paragraph.</p>
</body>

</html>
```

## HTML Text Fonts

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

### Example

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

```
<body>
  <h1 style="font-family:verdana">This is a heading</h1>
  <p style="font-family:courier">This is a paragraph.</p>
</body>

</html>
```



The `<font>` tag, supported in older versions of HTML, is not valid in HTML5.

---

## HTML Text Size

The **font-size** property defines the text size to be used for an HTML element:

### Example

```
<!DOCTYPE html>
<html>

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

</html>
```

## HTML Text Alignment

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

### Example

```
<!DOCTYPE html>
<html>

<body>
  <h1 style="text-align:center">Centered Heading</h1>
  <p>This is a paragraph.</p>
</body>

</html>
```



The `<center>` tag, supported in older versions of HTML, is not valid in HTML5.

## Chapter Summary

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

## HTML Text Formatting Elements

### HTML Formatting Elements

In the previous chapter, you learned about HTML **styling**, using the HTML **style attribute**.

HTML also defines special **elements**, for defining text with a special **meaning**.

HTML uses elements like `<b>` and `<i>` for formatting output, like **bold** or *italic* text.

Formatting elements were designed to display special **types of text**:

- Bold text
- Important text
- Italic text
- Emphasized text
- Marked text
- Small text
- Deleted text
- Inserted text
- Subscripts
- Superscripts

---

## HTML Bold and Strong Formatting

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

### Example

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

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

The HTML `<strong>` element defines **strong** text, with added semantic "strong" importance.

### Example

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

```
<p><strong>This text is strong</strong>.</p>
```

## HTML *Italic* and *Emphasized* Formatting

The HTML `<i>` element defines *italic* text, without any extra importance.

### Example

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

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

The HTML `<em>` element defines *emphasized* text, with added semantic importance.

### Example

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

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

Browsers display `<strong>` as `<b>`, and `<em>` as `<i>`.

However, there is a difference in the meaning of these tags: `<b>` and `<i>` defines bold and italic text, but `<strong>` and `<em>` means that the text is "important".

## HTML Small Formatting

The HTML `<small>` element defines **small** text:

### Example

```
<h2>HTML <small>Small</small> Formatting</h2>
```

## HTML Marked Formatting

The HTML `<mark>` element defines **marked** or highlighted text:

### Example

```
<h2>HTML <mark>Marked</mark> Formatting</h2>
```

## HTML Deleted Formatting

The HTML `<del>` element defines **deleted** (removed) text.

## Example

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

## HTML Subscript Formatting

The HTML **<sub>** element defines **subscripted** text.

## Example

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

## HTML Superscript Formatting

The HTML **<sup>** element defines **superscripted** text.

## Example

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

## HTML Comments

Comment tags **<!--** and **-->** are used to insert comments in HTML.

---

## HTML Comment Tags

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

## Example

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



**Note:** There is an exclamation point (!) in the opening tag, but not in the closing tag.

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

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

## Example

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

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

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

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

## Example

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

-->
```

# 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.

A hyperlink is an element, a text, or an image that you can click on, and jump to another document.

---

## HTML Links - Syntax

In HTML, links are defined with the **<a>** tag:

### Link Syntax:

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

### Example:

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

The **href** attribute specifies the destination address (<http://www.w3schools.com/html/>)

The **link text** is the visible part (Visit our HTML tutorial).



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



The link text does not have to be text. It can be an HTML image or any other HTML element.

---

## Local Links

The example above used an absolute URL (A full web address).

A local link (link to the same web site) is specified with a relative URL (without http://www....).

### Example:

```
<a href="html_images.asp">HTML Images</a>
```

## HTML Links - Colors and Icons

When you move the mouse cursor over a link, two things will normally happen:

- The mouse arrow will turn into a little hand
- The color of the link element will change

By default, links will appear as 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 defaults, using styles:

### Example

```
<style>
a:link      {color:#000000; background-color:transparent; text-decoration:none}
a:visited   {color:#000000; background-color:transparent; text-decoration:none}
a:hover     {color:#ff0000; background-color:transparent; text-decoration:underline}
a:active    {color:#ff0000; background-color:transparent; text-decoration:underline}
```

## HTML Links - The target Attribute

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

This example will open the linked document in a new browser window or in a new tab:

### Example

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

Target Value	Description
<code>_blank</code>	Opens the linked document in a new window or tab
<code>_self</code>	Opens the linked document in the same frame as it was clicked (this is default)
<code>_parent</code>	Opens the linked document in the parent frame
<code>_top</code>	Opens the linked document in the full body of the window
<code>framename</code>	Opens the linked document in a named frame

If your webpage is locked in a frame, you can use `target="_top"` to break out of the frame:

### Example


```
<a href="http://www.w3schools.com/html/" target="_top">HTML5 tutorial!</a>
```

## HTML Links - Image as Link

It is common to use images as links:

### Example

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

 `border:0` is added to prevent IE9 (and earlier) from displaying a border around the image.

## HTML Links - The id Attribute

The **id** attribute can be used to create bookmarks inside HTML documents.

Bookmarks are not displayed in any special way. They are invisible to the reader.

### Example

Add an id attribute to any `<a>` element:


```
<a id="tips">Useful Tips Section</a>
```

Then create a link to the `<a>` element (Useful Tips Section):

```
<a href="#tips">Visit the Useful Tips Section</a>
```

Or, create a link to the `<a>` element (Useful Tips Section) from another page:

```
<a href="http://www.w3schools.com/html_links.htm#tips">Visit the Useful Tips Section</a>
```

 Without a trailing slash on subfolder addresses, you might generate two requests to the server. Many servers will automatically add a slash to the address, and then create a request.

---

## Chapter Summary

- Use the HTML **<a>** element to define a link
- Use the HTML **href** attribute to define the link address
- Use the HTML **target** attribute to define where to open the linked document
- Use the HTML **<img>** element (inside `<a>`) to use an image as a link
- Use the HTML **id** attribute (`id=value`) to define bookmarks in a page
- Use the HTML **href** attribute (`href="#value"`) to address the bookmark

## HTML Images

```
<!DOCTYPE html>
<html>

<body>
  <h2>Spectacular Mountains</h2>
  
</body>

</html>
```

Always specify image size. If the size is unknown, the page will flicker while the image loads.

---

## HTML Images Syntax

In HTML, images are defined with the **<img>** tag.

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

The **src** attribute defines the url (web address) of the image:

```

```

---

## The alt Attribute

The **alt** attribute specifies an alternate text for the image, if it cannot be displayed.

The value of the alt attribute should describe the image in words:

### Example

```

```

The alt attribute is **required**. A web page will not validate correctly without it.

---

## HTML Screen Readers

Screen readers are software programs that can read what is displayed on a screen.

Used on the web, screen readers can "reproduce" HTML as text-to-speech, sound icons, or braille output.

Screen readers are used by people who are blind, visually impaired, or learning disabled.



Screen readers can read the **alt** attribute.

---

## Image Size - Width and Height

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

The values are specified in pixels (use px after the value):

### Example

```

```

Alternatively, you can use width and height **attributes**.

The values are specified in pixels (without px after the value):

## Example

```

```

## Width and Height or Style?

Both the width, the height, and the style attributes, are valid in the latest HTML5 standard.

We suggest you use the style attribute. It prevents styles sheets from changing the default size of images:

## Example

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

<body>




</body>

</html>
```



At W3schools we prefer to use the style attribute.

---

## Images in Another Folder

If not specified, the browser expects to find the image in the same folder as the web page.

However, it is common on the web, to store images in a sub-folder, and refer to the folder in the image name:

## Example

```

```

## HTML Tables

# HTML Table Example

Number	First Name	Last Name	Points
1	Eve	Jackson	94
2	John	Doe	80
3	Adam	Johnson	67
4	Jill	Smith	50

---

## Defining HTML Tables

### Example

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

Example explained:

Tables are defined with the **<table>** tag.

Tables are divided into **table rows** with the **<tr>** tag.

Table rows are divided into **table data** with the **<td>** tag.

A table row can also be divided into **table headings** with the **<th>** tag.



Table data **<td>** are the data containers of the table.  
They can contain all sorts of HTML elements like text, images, lists, other tab etc.

## An HTML Table with a Border Attribute

If you do not specify a border for the table, it will be displayed without borders.

A border can be added using the border attribute:

### Example

```
<table border="1" style="width:100%">
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

To add borders, use the **CSS border** property:

### Example

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

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

---

## An HTML Table with Collapsed Borders

If you want the borders to collapse into one border, add **CSS border-collapse**:

### Example

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

## An HTML Table with Cell Padding

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

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

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

### Example

```
table, th, td {  
    border: 1px solid black;  
    border-collapse: collapse;  
}  
th,td {  
    padding: 15px;  
}
```

## HTML Table Headings

Table headings are defined with the **<th>** tag.

By default, all major browsers display table headings as bold and centered:

### Example

```
<table style="width:100%">  
  <tr>  
    <th>Firstname</th>  
    <th>Lastname</th>  
    <th>Points</th>  
  </tr>  
  <tr>  
    <td>Eve</td>  
    <td>Jackson</td>  
    <td>94</td>  
  </tr>  
</table>
```

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

### Example

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

## An HTML Table with Border Spacing

Border spacing specifies the space between the cells.

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

### Example

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



## Table Cells that Span Many Columns

To make a cell span more than one column, use the **colspan** attribute:

### Example

```
<table style="width:100%">
  <tr>
    <th>Name</th>
    <th colspan="2">Telephone</th>
  </tr>
  <tr>
    <td>Bill Gates</td>
    <td>555 77 854</td>
    <td>555 77 855</td>
  </tr>
</table>
```

## Table Cells that Span Many Rows

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

### Example

```
<table style="width:100%">
  <tr>
    <th>Name:</th>
    <td>Bill Gates</td>
  </tr>
  <tr>
    <th rowspan="2">Telephone:</th>
    <td>555 77 854</td>
  </tr>
  <tr>
    <td>555 77 855</td>
  </tr>
</table>
```

## An HTML Table With a Caption

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>

```

## Chapter Summary

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

## HTML **<div>** Tag

### Example

A section in a document that will be displayed in blue:

```

<div style="color:#0000FF">
    <h3>This is a heading</h3>
    <p>This is a paragraph.</p>
</div>

```

## Definition and Usage

The **<div>** tag defines a division or a section in an HTML document.

The **<div>** tag is used to group block-elements to format them with CSS.

## Tips and Notes

**Tip:** The **<div>** element is very often used together with CSS, to layout a web page.

**Note:** By default, browsers always place a line break before and after the **<div>** element. However, this can be changed with CSS.

# Differences Between HTML 4.01 and HTML5

The align attribute not supported in HTML5.

---

## Attributes

Attribute	Value	Description
<u>align</u>	left right center justify	<b>Not supported in HTML5.</b> Specifies the alignment of the content inside a <div> element

---

## Global Attributes

The <div> tag also supports the Global Attributes in HTML.

## Default CSS Settings

Most browsers will display the <div> element with the following default values:

### Example

```
div {  
    display: block;  
}
```

## CSS position Property

### Example

Position an <h2> element:

```
h2 {  
    position: absolute;  
    left: 100px;  
    top: 150px;  
}
```

# CSS Syntax

```
position: static|absolute|fixed|relative|initial|inherit;
```

## Property Values

Value	Description
static	Default value. Elements render in order, as they appear in the document flow
absolute	The element is positioned relative to its first positioned (not static) ancestor element
fixed	The element is positioned relative to the browser window
relative	The element is positioned relative to its normal position, so "left:20" 20 pixels to the element's LEFT position
initial	Sets this property to its default value. <a href="#">Read about <i>initial</i></a>
inherit	Inherits this property from its parent element. <a href="#">Read about <i>inherit</i></a>

## Example

How to position an element relative to its normal position:

```
h2.pos_left {  
    position: relative;  
    left: -20px;  
}  
  
h2.pos_right {  
    position: relative;  
    left: 20px;  
}
```

## CSS Margin

The CSS margin properties define the space around elements.

---

## Margin

The margin clears an area around an element (outside the border). The margin does not have a background color, and is completely transparent.

The top, right, bottom, and left margin can be changed independently using separate properties. A shorthand margin property can also be used, to change all margins at once.

## Possible Values

Value	Description
auto	The browser calculates a margin
<i>length</i>	Specifies a margin in px, pt, cm, etc. Default value is 0px
%	Specifies a margin in percent of the width of the containing element
inherit	Specifies that the margin should be inherited from the parent element



**Note:** It is also possible to use negative values, to overlap content.

---

## Margin - Individual sides

In CSS, it is possible to specify different margins for different sides of an element:

### Example

```
p {  
  margin-top: 100px;  
  margin-bottom: 100px;  
  margin-right: 150px;  
  margin-left: 50px;  
}
```

The margin property can have from one to four values.

- **margin: 25px 50px 75px 100px;**

- top margin is 25px
  - right margin is 50px
  - bottom margin is 75px
  - left margin is 100px
- 
- **margin: 25px 50px 75px;**
    - top margin is 25px
    - right and left margins are 50px
    - bottom margin is 75px
- 
- **margin: 25px 50px;**
    - top and bottom margins are 25px
    - right and left margins are 50px
- 
- **margin: 25px;**
    - all four margins are 25px

## CSS Padding

The CSS padding properties define the space between the element border and the element content.

---

### Padding

The padding clears an area around the content (inside the border) of an element. The padding is affected by the background color of the element.

The top, right, bottom, and left padding can be changed independently using separate properties. A shorthand padding property can also be used, to change all paddings at once.

### Possible Values

Value	Description
<i>length</i>	Defines a fixed padding (in pixels, pt, em, etc.)

%	Defines a padding in % of the containing element
---	--

---

## Padding - Individual sides

In CSS, it is possible to specify different padding for different sides:

### Example

```
p {  
  padding-top: 25px;  
  padding-right: 50px;  
  padding-bottom: 25px;  
  padding-left: 50px;  
}
```

## Padding - Shorthand property

To shorten the code, it is possible to specify all the padding properties in one property. This is called a shorthand property.

The shorthand property for all the padding properties is "padding":

### Example

```
p {  
  padding: 25px 50px;  
}
```

The padding property can have from one to four values.

- **padding: 25px 50px 75px 100px;**
  - top padding is 25px
  - right padding is 50px
  - bottom padding is 75px
  - left padding is 100px
- **padding: 25px 50px 75px;**
  - top padding is 25px
  - right and left paddings are 50px
  - bottom padding is 75px
- **padding: 25px 50px;**

- top and bottom paddings are 25px
  - right and left paddings are 50px
- **padding: 25px;**
  - all four paddings are 25px



# CSS Tutorial

Save a lot of work with CSS!

In our CSS tutorial you will learn how to use CSS to control the style and layout of multiple Web pages all at once.

## Examples in Each Chapter

This CSS tutorial contains hundreds of CSS examples.

With our online editor, you can edit the CSS, and click on a button to view the result.

### CSS Example

```
body {  
    background-color: #d0e4fe;  
}  
  
h1 {  
    color: orange;  
    text-align: center;  
}  
  
p {  
    font-family: "Times New Roman";  
    font-size: 20px;  
}
```

## CSS Introduction

### What is CSS?

- **CSS** stands for **Cascading Style Sheets**
- Styles define **how to display** HTML elements
- Styles were added to HTML 4.0 **to solve a problem**
- **External Style Sheets** can save a lot of work
- External Style Sheets are stored in **CSS files**

### Styles Solved a Big Problem

HTML was never intended to contain tags for formatting a document.

HTML was intended to define the content of a document, like:

```
<h1>This is a heading</h1>
```

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

When tags like `<font>`, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large web sites, where fonts and color information were added to every single page, became a long and expensive process.

To solve this problem, the World Wide Web Consortium (W3C) created CSS.

In HTML 4.0, all formatting could be removed from the HTML document, and stored in a separate CSS file.

All browsers support CSS today.

## CSS Saves a Lot of Work!

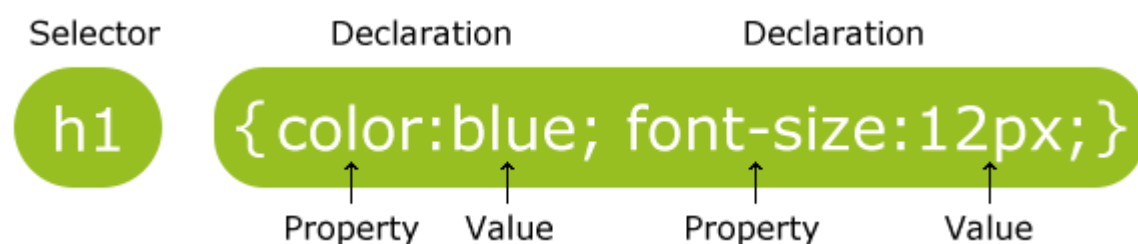
CSS defines HOW HTML elements are to be displayed.

Styles are normally saved in external .css files. External style sheets enable you to change the appearance and layout of all the pages in a Web site, just by editing one single file!

## CSS Syntax

### CSS Syntax

A CSS rule set consists of a selector and a declaration block:



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a property name and a value, separated by a colon.

---

## CSS Example

A CSS declaration always ends with a semicolon, and declaration groups are surrounded by curly braces:

```
p {color:red;text-align:center;}
```

To make the CSS code more readable, you can put one declaration on each line, like this:

## Example

```
p {  
    color: red;  
    text-align: center;  
}
```

## CSS Comments

Comments are used to explain your code, and may help you when you edit the source code at a later date. Comments are ignored by browsers.

A CSS comment starts with `/*` and ends with `*/`. Comments can also span multiple lines:

## Example

```
p {  
    color: red;  
    /* This is a single-line comment */  
    text-align: center;  
}  
  
/* This is  
a multi-line  
comment */
```

## CSS Selectors

### CSS Selectors

CSS selectors allow you to select and manipulate HTML element(s).

CSS selectors are used to "find" (or select) HTML elements based on their id, classes, types, attributes, values of attributes and much more.

---

## The element Selector

The element selector selects elements based on the element name.

You can select all `<p>` elements on a page like this: (all `<p>` elements will be center-aligned, with a red text color)

## Example

```
p {  
    text-align: center;  
    color: red;  
}
```

# The id Selector

The id selector uses the id attribute of an HTML tag to find the specific element.

An id should be unique within a page, so you should use the id selector when you want to find a single, unique element.

To find an element with a specific id, write a hash character, followed by the id of the element.

The style rule below will be applied to the HTML element with id="para1":

## Example

```
#para1 {  
    text-align: center;  
    color: red;  
}
```



Do **NOT** start an ID name with a number!

---

# The class Selector

The class selector finds elements with the specific class.

The class selector uses the HTML class attribute.

To find elements with a specific class, write a period character, followed by the name of the class:

In the example below, all HTML elements with class="center" will be center-aligned:

## Example

```
.center {  
    text-align: center;  
    color: red;  
}
```

You can also specify that only specific HTML elements should be affected by a class.

In the example below, all p elements with class="center" will be center-aligned:

## Example

```
p.center {  
  text-align: center;  
  color: red;  
}
```



Do **NOT** start a class name with a number!

## Grouping Selectors

In style sheets there are often elements with the same style:

```
h1 {  
  text-align: center;  
  color: red;  
}  
  
h2 {  
  text-align: center;  
  color: red;  
}  
  
p {  
  text-align: center;  
  color: red;  
}
```

To minimize the code, you can group selectors.

To group selectors, separate each selector with a comma.

In the example below we have grouped the selectors from the code above:

## Example

```
h1, h2, p {  
  text-align: center;  
  color: red;  
}
```

## CSS How To...

When a browser reads a style sheet, it will format the document according to the information in the style sheet.

---

## Three Ways to Insert CSS

There are three ways of inserting a style sheet:

- External style sheet
- Internal style sheet
- Inline style

---

## External Style Sheet

An external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing just one file.

Each page must include a link to the style sheet with the <link> tag. The <link> tag goes inside the head section:

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

An external style sheet can be written in any text editor. The file should not contain any html tags. The style sheet file must be saved with a .css extension. An example of a style sheet file called "myStyle.css", is shown below:

```
body {
    background-color: lightblue;
}
h1 {
    color: navy;
    margin-left: 20px;
}
```



Do not add a space between the property value and the unit (such as margin-left: 20 px;).  
margin-left: 20px;

# Internal Style Sheet

An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section of an HTML page, inside the `<style>` tag, like this:

## Example

```
<head>
<style>
body {
    background-color: linen;
}
h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
```

# Inline Styles

An inline style loses many of the advantages of a style sheet (by mixing content with presentation). Use this method sparingly!

To use inline styles, add the style attribute to the relevant tag. The style attribute can contain any CSS property. The example shows how to change the color and the left margin of a h1 element:

## Example

```
<h1 style="color:blue;margin-left:30px;">This is a heading.</h1>
```

# Multiple Style Sheets

If some properties have been set for the same selector in different style sheets, the values will be inherited from the more specific style sheet.

For example, assume that an external style sheet has the following properties for the h1 selector:

```
h1 {
    color: navy;
    margin-left: 20px;
}
```

then, assume that an internal style sheet also has the following property for the h1 selector:

```
h1 {
    color: orange;
}
```

If the page with the internal style sheet also links to the external style sheet the properties for the h1 element will be:

```
color: orange;
margin-left: 20px;
```

The left margin is inherited from the external style sheet and the color is replaced by the internal style sheet.

---

## Multiple Styles Will Cascade into One

Styles can be specified:

- inside an HTML element
- inside the head section of an HTML page
- in an external CSS file

**Tip:** Even multiple external style sheets can be referenced inside a single HTML document.

### Cascading order

What style will be used when there is more than one style specified for an HTML element?

Generally speaking we can say that all the styles will "cascade" into a new "virtual" style sheet by the following rules, where number four has the highest priority:

1. Browser default
2. External style sheet
3. Internal style sheet (in the head section)
4. Inline style (inside an HTML element)
5. o, an inline style (inside an HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet, or in a browser (a default value).



**Note:** If the link to the external style sheet is placed after the internal style sheet in HTML <head>, the external style sheet will override the internal sheet!

## CSS Background



CSS background properties are used to define the background effects of an element.

CSS properties used for background effects:

- background-color
- background-image
- background-repeat
- background-attachment
- background-position



## Background Color

The background-color property specifies the background color of an element.

The background color of a page is defined in the body selector:

### Example

```
body {  
    background-color: #b0c4de;  
}
```

With CSS, a color is most often specified by:

- a HEX value - like "#ff0000"
- an RGB value - like "rgb(255,0,0)"
- a color name - like "red"

Look at [CSS Color Values](#) for a complete list of possible color values.

In the example below, the h1, p, and div elements have different background colors:

### Example

```
h1 {  
    background-color: #6495ed;  
}  
  
p {  
    background-color: #e0ffff;  
}  
  
div {  
    background-color: #b0c4de;  
}
```

## Background Image

The background-image property specifies an image to use as the background of an element.

By default, the image is repeated so it covers the entire element.

The background image for a page can be set like this:

### Example

```
body {  
    background-image: url("paper.gif");  
}
```

## Background Image - Repeat Horizontally or Vertically

If the image is repeated only horizontally (repeat-x), the background will look better:

### Example

```
body {  
    background-image: url("gradient_bg.png");  
    background-repeat: repeat-x;  
}
```

## Background Image - Set position and no-repeat



**Note:** When using a background image, use an image that does not disturb the text.

Showing the image only once is specified by the background-repeat property:

### Example

```
body {  
    background-image: url("img_tree.png");  
    background-repeat: no-repeat;  
}
```

In the example above, the background image is shown in the same place as the text. We want to change the position of the image, so that it does not disturb the text too much.

The position of the image is specified by the background-position property:

### Example

```
body {  
    background-image: url("img_tree.png");  
    background-repeat: no-repeat;  
    background-position: right top;  
}
```

## Background - Shorthand property

As you can see from the examples above, there are many properties to consider when dealing with backgrounds.

To shorten the code, it is also possible to specify all the properties in one single property. This is called a shorthand property.

The shorthand property for background is simply "background":

### Example

```
body {  
    background: #ffffff url("img_tree.png") no-repeat right top;  
}
```

When using the shorthand property the order of the property values is:

- background-color
- background-image
- background-repeat
- background-attachment
- background-position

It does not matter if one of the property values is missing, as long as the ones that are present are in this order.

# HTML Forms

## The <form> Element

HTML forms are used to collect user input.

The **<form>** element defines an HTML form:

### Example

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

HTML forms contain **form elements**.

Form elements are different types of input elements, checkboxes, radio buttons, submit buttons, and more.

## The <input> Element

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

The <input> element has many variations, depending on the **type** attribute.

Here are the types used in this chapter:

Type	Description
text	Defines normal text input
radio	Defines radio button input (for selecting one of many choices)
submit	Defines a submit button (for submitting the form)
You will learn a lot more about input types later in this tutorial.	

# Text Input

**<input type="text">** defines a one-line input field for **text input**:

```
<form>
First name:<br>
<input type="text" name="firstname">
<br>
Last name:<br>
<input type="text" name="lastname">
</form>
```

This is how it will look like in a browser:

First name:

Last name:

**Note:** The form itself is not visible. Also note that the default width of a text field is 20 characters.

# Radio Button Input

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

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

```
<form>
<input type="radio" name="sex" value="male" checked>Male
<br>
<input type="radio" name="sex" value="female">Female
</form>
```

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

- ☒ Male
- ☐ Female

# The Submit Button

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

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

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

```
<form action="action_page.php">
First name:<br>
<input type="text" name="firstname" value="Mickey">
<br>
Last name:<br>
<input type="text" name="lastname" value="Mouse">
<br><br>
<input type="submit" value="Submit">
</form>
```

First name:

Last name:

# The Action Attribute

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

The common way to submit a form to a server, is by using a submit button.

Normally, the form is submitted to a web page on a web server.

In the example above, a server-side script is specified to handle the submitted form:

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

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

# The Method Attribute

The **method attribute** specifies the HTTP method (**GET** or **POST**) to be used when submitting the forms:

```
<form action="action_page.php" method="GET">
```

or:

```
<form action="action_page.php" method="POST">
```

## When to Use GET?

You can use GET (the default method):

If the form submission is passive (like a search engine query), and without sensitive information.

When you use GET, the form data will be visible in the page address:

action\_page.php?firstname=Mickey&lastname=Mouse

GET is best suited to short amounts of data. Size limitations are set in your browser.

## When to Use POST?

You should use POST:

If the form is updating data, or includes sensitive information (password).

POST offers better security because the submitted data is not visible in the page address.

## The Name Attribute

To be submitted correctly, each input field must have a name attribute.

This example will only submit the "Last name" input field:

```
<form action="action_page.php">  
First name:<br>
```

```

<input type="text" value="Mickey">
<br>
Last name:<br>
<input type="text" name="lastname" value="Mouse">
<br><br>
<input type="submit" value="Submit">
</form>

```

## Grouping Form Data with <fieldset>

The **<fieldset>** element groups related data in a form.

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

```

<form action="action_page.php">
<fieldset>
<legend>Personal information:</legend>
First name:<br>
<input type="text" name="firstname" value="Mickey">
<br>
Last name:<br>
<input type="text" name="lastname" value="Mouse">
<br><br>
<input type="submit" value="Submit"></fieldset>
</form>

```

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

Personal information:First name:

Last name:



## HTML Form Attributes

An HTML <form> element, with all possible attributes set, will look like this:

```

<form action="action_page.php" method="GET" target="_blank" accept-
charset="UTF-8"
enctype="application/x-www-form-

```



```
urlencoded" autocomplete="off" novalidate>
.
form elements
.
</form>
```

Here is the list of <form> attributes:

Attribute	Description
accept-charset	Specifies the charset used in the submitted form (default: the page charset).
action	Specifies an address (url) where to submit the form (default: the submitting page).
autocomplete	Specifies if the browser should autocomplete the form (default: on).
enctype	Specifies the encoding of the submitted data (default: is url-encoded).
method	Specifies the HTTP method used when submitting the form (default: GET).
name	Specifies a name used to identify the form (for DOM usage: document.forms.name).
novalidate	Specifies that the browser should not validate the form.
target	Specifies the target of the address in the action attribute (default: _self).
You will learn more about attributes in the next chapters.	

# HTML Form Elements

This chapter describes all HTML form elements.

## The `<input>` Element

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

The `<input>` element can vary in many ways, depending on the **type** attribute.

## The `<select>` Element (Drop-Down List)

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

```
<select name="cars">
<option value="volvo">Volvo</option>
<option value="saab">Saab</option>
<option value="fiat">Fiat</option>
<option value="audi">Audi</option>
</select>
```

The `<option>` elements defines the options to select.

The list will normally show the first item as selected.

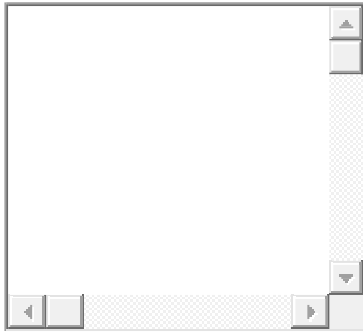
You can add a `selected` attribute to define a predefined option.

## The `<textarea>` Element

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

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

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



## The <button> Element

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

```
<button type="button"> Click Me!</button>
```

## HTML5 Form Elements

HTML5 added the following form elements:

- <datalist>
- <keygen>
- <output>

By default, browsers do not display unknown elements. New elements will not destroy your page.

## HTML Form Elements

= new in HTML5.

Tag	Description
<u>&lt;form&gt;</u>	Defines an HTML form for user input
<u>&lt;input&gt;</u>	Defines an input control

<u>&lt;textarea&gt;</u>	Defines a multiline input control (text area)
<u>&lt;label&gt;</u>	Defines a label for an <input> element
<u>&lt;fieldset&gt;</u>	Groups related elements in a form
<u>&lt;legend&gt;</u>	Defines a caption for a <fieldset> element
<u>&lt;select&gt;</u>	Defines a drop-down list
<u>&lt;optgroup&gt;</u>	Defines a group of related options in a drop-down list
<u>&lt;option&gt;</u>	Defines an option in a drop-down list
<u>&lt;button&gt;</u>	Defines a clickable button
<u>&lt;datalist&gt;</u>	Specifies a list of pre-defined options for input controls
<u>&lt;keygen&gt;</u>	Defines a key-pair generator field (for forms)
<u>&lt;output&gt;</u>	Defines the result of a calculation

# Input Types

This chapter describes the input types of the `<input>` element.

## Input Type: text

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

```
<form>
First name:<br>
<input type="text" name="firstname">
<br>
Last name:<br>
<input type="text" name="lastname">
</form>
```

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

First name:

Last Name:

## Input Type: password

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

```
<form>
User name:<br>
<input type="text" name="username">
<br>
User password:<br>
<input type="password" name="psw">
</form>
```

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

First name:

Password:

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

## Input Type: submit

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

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

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

```
<form action="action_page.php">
First name:<br>
<input type="text" name="firstname" value="Mickey">
<br>
Last name:<br>
<input type="text" name="lastname" value="Mouse">
<br><br>
<input type="submit" value="Submit">
</form>
```

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

First name:

Last name:

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

```
<form action="action_page.php">
First name:<br>
<input type="text" name="firstname" value="Mickey">
<br>
Last name:<br>
<input type="text" name="lastname" value="Mouse">
<br><br>
<input type="submit">
</form>
```

# Input Type: radio

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

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

```
<form>
<input type="radio" name="sex" value="male" checked>Male
<br>
<input type="radio" name="sex" value="female">Female
</form>
```

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

- ☒ Male
- ☐ Female

# Input Type: checkbox

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

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

```
<form>
<input type="checkbox" name="vehicle" value="Bike">I have a bike
<br>
<input type="checkbox" name="vehicle" value="Car">I have a car
</form>
```

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

- ☐ I have a bike
- ☐ I have a car

# Input Type: button

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

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

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

# HTML5 Input Types

HTML5 added several new input types:

- color
- date
- datetime
- datetime-local
- email
- month
- number
- range
- search
- tel
- time
- url
- week

Input types, not supported by old web browsers, will behave as input type text.

## Input Type: number

The **<input type="number">** is used for input fields that should contain a numeric value.

You can set restrictions on the numbers.

Depending on browser support, the restrictions can apply to the input field.

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

## Input Restrictions

Here is a list of some common input restrictions (some are new in HTML5):



Attribute	Description
disabled	Specifies that an input field should be disabled
max	Specifies the maximum value for an input field
maxlength	Specifies the maximum number of character for an input field
min	Specifies the minimum value for an input field
pattern	Specifies a regular expression to check the input value against
readonly	Specifies that an input field is read only (cannot be changed)
required	Specifies that an input field is required (must be filled out)
size	Specifies the width (in characters) of an input field
step	Specifies the legal number intervals for an input field
value	Specifies the default value for an input field