HTML Introduction

HTML is the standard markup language for creating Web pages.

What is HTML?

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

A Simple HTML Document

Example

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

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

</body>
</html>
```

Example Explained

- The <!DOCTYPE html> declaration defines that this document is an HTML5 document

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

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

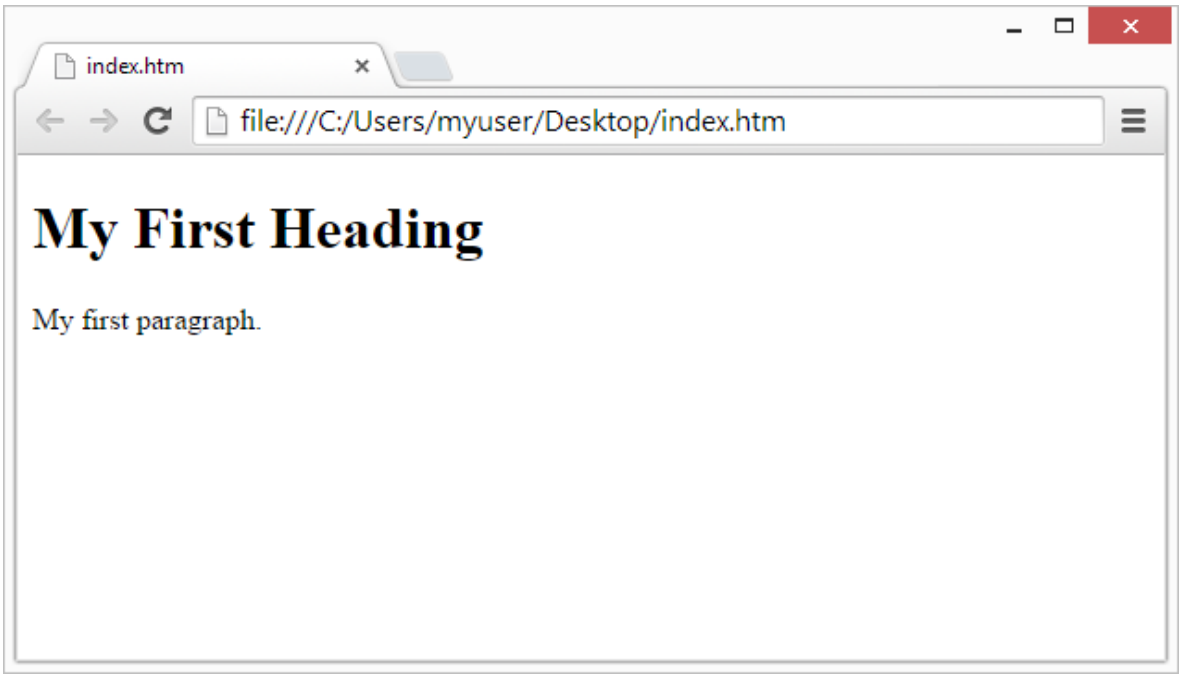
Note: Some HTML elements have no content (like the
 element). These elements are called empty elements. Empty elements do not have an end tag.

Web Browsers

The purpose of a web browser (Chrome, Edge, Firefox, Safari) is to read

HTML documents and display them correctly.

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



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
2014	W3C Recommendation: HTML5
2016	W3C Candidate Recommendation: HTML 5.1
2017	W3C Recommendation: HTML5.1 2nd Edition
2017	W3C Recommendation: HTML5.2

HTML Editors:

A simple text editor is all you need to learn HTML.

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 2: Write Some HTML

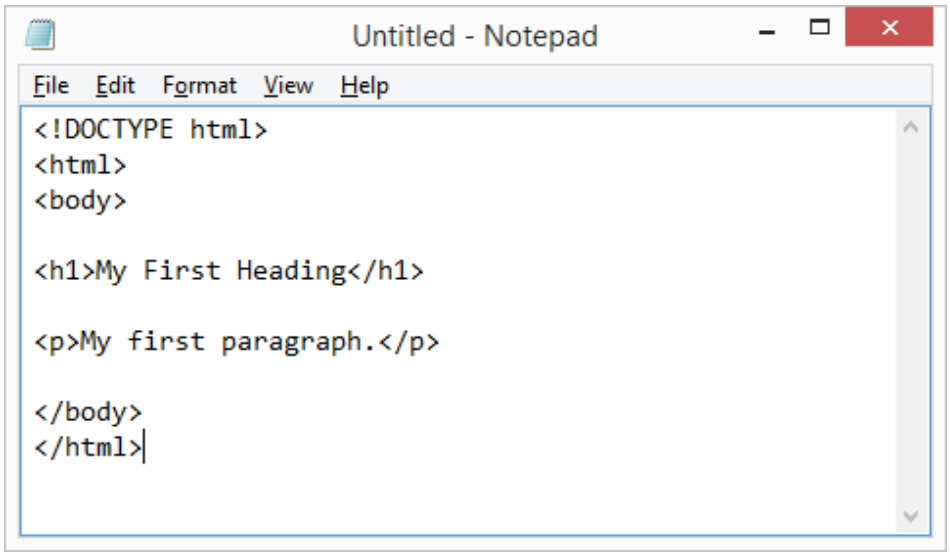
Write or copy the following HTML code into Notepad:

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

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```

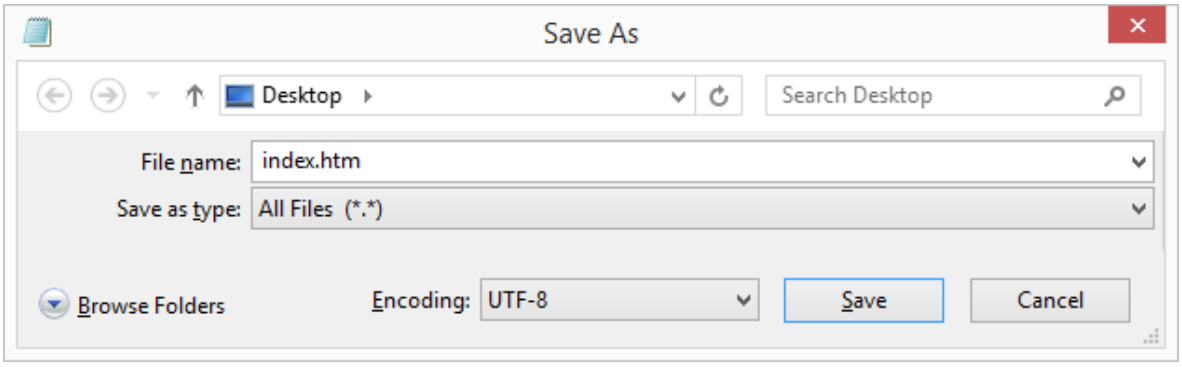


Step 3: Save the HTML Page

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

Name the file **"index.htm"** and set the encoding to **UTF-8** (which is the

preferred encoding for HTML files).

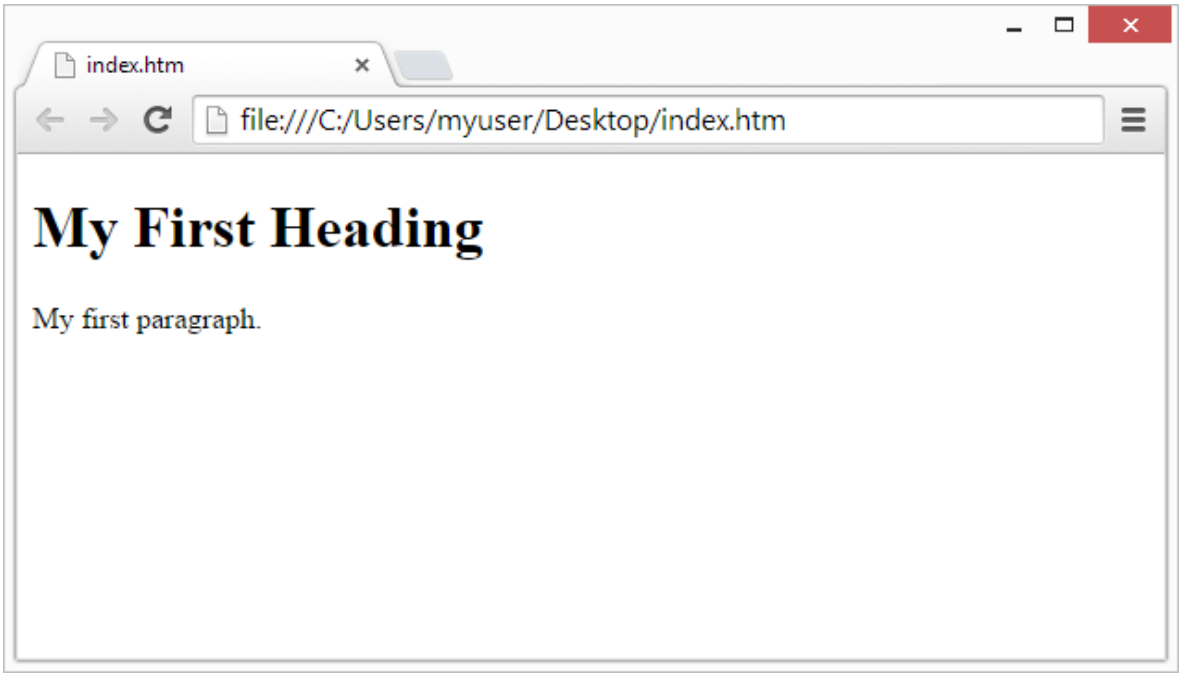


Tip: You can use either .htm or .html as file extension. There is no difference; it is up to you.

Step 4: View the HTML Page in Your Browser

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

The result will look much like this:



HTML Basic Examples:

In this chapter we will show some basic HTML examples.

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

HTML Documents

All HTML documents must start with a document type declaration: <!DOCTYPE html>.

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

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

Example

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

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

</body>
</html>
```

The <!DOCTYPE> Declaration

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

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

The <!DOCTYPE> declaration is not case sensitive.

The <!DOCTYPE> declaration for HTML5 is:

<!DOCTYPE html>

HTML Headings

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

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

Example

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

HTML Paragraphs

HTML paragraphs are defined with the <p> tag:

Example

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

HTML Links

HTML links are defined with the <a> tag:

Example

This is a link
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="hello.jpg" alt="hello.com" width="104" height="142">

How to View HTML Source

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

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.

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.

Nested HTML Elements

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

All HTML documents consist of nested HTML elements.

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

Example

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

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

</body>
</html>
```

Example Explained

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

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

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

```
<body>

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

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

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

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

<h1>My First Heading</h1>
My first paragraph.
The <h1> element defines a heading.

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

<h1>My First Heading</h1>
The element defines a paragraph.

It has a start tag and an end tag :

My first paragraph.

Never Skip the End Tag

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

Example

<html>
<body>

This is a paragraph
This is a paragraph

</body>
</html>
However, never rely on this! Unexpected results and errors may occur

if you forget the end tag!

Empty HTML Elements

HTML elements with no content are called empty elements.

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

Example

This is a
 paragraph with a line break.

HTML is Not Case Sensitive

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

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

Visit hello

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">
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.hello.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):

Example

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

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">

Example

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

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

The style Attribute

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

Example

This is a red paragraph.

You will learn more about styles in our HTML Styles chapter.

The lang Attribute

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

The following example specifies English as the language:

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

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

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

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

You can see all the language codes in our HTML Language Code Reference.

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

This is a paragraph.

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.

At hello we always use lowercase attribute names.

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:

Visit our HTML tutorial

Bad:

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

Example

<p title=About hello>
At hello we always use quotes around attribute values.

Single or Double Quotes?

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

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

<p title='John "ShotGun" Nelson'>
Or vice versa:

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

Chapter Summary

- All HTML elements can have **attributes**
- The href attribute of <a> specifies the URL of the page the link goes to
- The src attribute of <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

HTML Headings

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

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

Example

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

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

Headings Are Important

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

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

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

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

Bigger Headings

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

Example

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

HTML Paragraphs

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

HTML Paragraphs

The HTML <p> element defines a paragraph.

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

Example

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

HTML Display

You cannot be sure how HTML will be displayed.

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

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

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

Example

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

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

HTML Horizontal Rules

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

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

Example

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

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

HTML Line Breaks

The HTML
 element defines a line break.

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

Example

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

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

The Poem Problem

This poem will display on a single line:

Example

```
<p>
  My Bonnie lies over the ocean.

  My Bonnie lies over the sea.

  My Bonnie lies over the ocean.

  Oh, bring back my Bonnie to me.
</p>
```

Solution - The HTML <pre> Element

The HTML <pre> element defines preformatted text.

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

Example

```
<pre>
  My Bonnie lies over the ocean.

  My Bonnie lies over the sea.

  My Bonnie lies over the ocean.

  Oh, bring back my Bonnie to me.
</pre>
```

HTML Styles

The HTML style attribute is used to add styles to an element, such as

color, font, size, and more.

Example

I am Red

I am Blue

I am Big

The HTML Style Attribute

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

The HTML style attribute has the following syntax:

<*tagname* style="*property*:*value;*">
The ***property*** is a CSS property. The ***value*** is a CSS value.

You will learn more about CSS later in this tutorial.

Background Color

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

Example

Set the background color for a page to powderblue:

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

<h1>This is a heading</h1>
This is a paragraph.

</body>

Example

Set background color for two different elements:

```
<body>

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

</body>
```

Text Color

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

Example

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

Fonts

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

Example

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

Text Size

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

Example

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

Text Alignment

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

Example

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

Chapter Summary

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

HTML Text Formatting

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

Example

This text is bold

This text is italic

This is subscript and superscript

HTML Formatting Elements

Formatting elements were designed to display special types of text:

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

HTML and Elements

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

Example

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

Example

This text is important!

HTML <i> and Elements

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

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

Example

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

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

Example

This text is emphasized

HTML <small> Element

The HTML <small> element defines smaller text:

Example

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

HTML <mark> Element

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

Example

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

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:

Example

My favorite color is ~~blue~~ red.

HTML <ins> Element

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

Example

My favorite color is ~~blue~~ <ins>red</ins>.

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 H_2O:

Example

This is _{subscripted} text.

HTML <sup> Element

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

Example

This is ^{superscripted} 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

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

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

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

HTML <address> for Contact

Information

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

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

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

Example

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

HTML <cite> for Work Title

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

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

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

Example

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

HTML <bdo> for Bi-Directional Override

BDO stands for Bi-Directional Override.

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

Example

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

HTML Comments

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

HTML Comment Tag

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

<!-- Write your comments here -->
Notice that there is an exclamation point (!) in the start tag, but not in the end tag.

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

Add Comments

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

Example

<!-- This is a comment -->

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

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

Hide Content

Comments can be used to hide content.

This can be helpful if you hide content temporarily:

Example

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

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

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

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

Example

Hide a section of HTML code:

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

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

Hide Inline Content

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

Example

Hide a part of a paragraph:

This <!-- great text --> 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:

link text
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 hello.com:

Visit hello.com!
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:

Visit hello!

Absolute URLs vs. Relative

URLs

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

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

Example

```
<h2>Absolute URLs</h2>
<p><a href="https://www.w3.org/">W3C</a></p>
<p><a href="https://www.google.com/">Google</a></p>

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

HTML Links - Use an Image as a Link

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

Example

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

Link to an Email Address

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

Example

Send email

Button as a Link

To use an HTML button as a link, you have to add some JavaScript code.

JavaScript allows you to specify what happens at certain events, such as a click of a button:

Example

<button onclick="document.location='default.asp'">HTML Tutorial</
button>
Tip: Learn more about JavaScript in our JavaScript Tutorial.

Link Titles

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

Example

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

More on Absolute URLs and Relative URLs

Example

Use a full URL to link to a web page:

HTML tutorial

Example

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

HTML tutorial

Example

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

HTML tutorial

You can read more about file paths in the chapter HTML File Paths.

Chapter Summary

- Use the <a> element to define a link
- Use the href attribute to define the link address
- Use the target attribute to define where to open the linked document
- Use the <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>	Defines a hyperlink

HTML Images

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

Example

<img src="pic_trulli.jpg" alt="Italian Trulli">

Example

<img src="img_girl.jpg" alt="Girl in a jacket">

Example

<img src="img_chania.jpg" alt="Flowers in Chania">

HTML Images Syntax

The HTML <img> tag is used to embed an image in a web page.

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

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

The <img> tag has two required attributes:

- src - Specifies the path to the image
- alt - Specifies an alternate text for the image

Syntax

<img src="*url*" alt="*alternatetext*">

The src Attribute

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

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

Example

```
<img src="img_chania.jpg" alt="Flowers in Chania">
```

The alt Attribute

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

The value of the alt attribute should describe the image:

Example

```
<img src="img_chania.jpg" alt="Flowers in Chania">
```

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

Example

```
<img src="wrongname.gif" alt="Flowers in Chania">
```

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

Image Size - Width and Height

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

Example

<img src="img_girl.jpg" alt="Girl in a jacket"
style="width:500px;height:600px;">
Alternatively, you can use the width and height attributes:

Example

<img src="img_girl.jpg" alt="Girl in a jacket" width="500"
height="600">
The width and height attributes always define the width and height of the image in pixels.

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

Width and Height, or Style?

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

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

Example

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

<img src="html5.gif" alt="HTML5 Icon" width="128" height="128">

<img src="html5.gif" alt="HTML5 Icon"
```

style="width:128px;height:128px;">

</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.hello.com/images/hello_green.jpg"
alt="hello.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.

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

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:

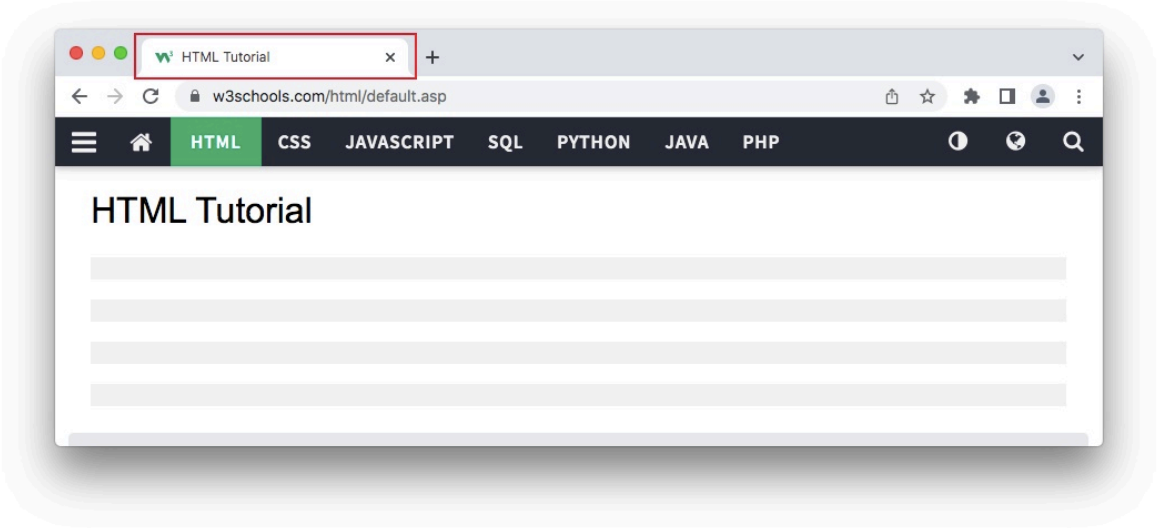
Example

```
<!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>	Defines the title of the document

HTML Tables

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

Example

Company		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>
```

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
	Defines an unordered list

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

HTML Forms

An HTML form is used to collect user input. The user input is most often sent to a server for processing.

Example

First name:
John
Last name:
Doe

The <form> Element

The HTML <form> element is used to create an HTML form for user input:

<form>
.
form elements
.
</form>
The <form> element is a container for different types of input elements, such as: text fields, checkboxes, radio buttons, submit buttons, etc.

All the different form elements are covered in this chapter: HTML Form Elements.

The <input> Element

The HTML <input> element is the most used form element.

An <input> element can be displayed in many ways, depending on the type attribute.

Here are some examples:

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

All the different input types are covered in this chapter: HTML Input Types.

Text Fields

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

Example

A form with input fields for text:

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

```
  <input type="text" id="lname" name="lname">
</form>
```
This is how the HTML code above will be displayed in a browser:

First name:

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

The <label> Element

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

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

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

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

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

Radio Buttons

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

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

Example

A form with radio buttons:

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

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

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

Choose your favorite Web language:

HTML
CSS
JavaScript

Checkboxes

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

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

Example

A form with checkboxes:

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

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

I have a bike
I have a car
I have a boat

The Submit Button

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

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

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

Example

A form with a submit button:

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

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

First name:
John
Last name:
Doe

The Name Attribute for <input>

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

If the name attribute is omitted, the value of the input field will not be sent

at all.

Example

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

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

HTML Form Attributes

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

The Action Attribute

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

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

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

Example

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

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

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

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

The Target Attribute

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

The target attribute can have one of the following values:

Value	Description
_blank	The response is displayed in a new window or tab
_self	The response is displayed in the current window
_parent	The response is displayed in the parent frame
_top	The response is displayed in the full body of the window
framename	The response is displayed in a named iframe

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

Example

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

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

The Method Attribute

The method attribute specifies the HTTP method to be used when

submitting the form data.

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

The default HTTP method when submitting form data is GET.

Example

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

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

Example

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

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

Notes on GET:

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

Notes on POST:

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

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

The Autocomplete Attribute

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

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

Example

A form with autocomplete on:

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

The Novalidate Attribute

The novalidate attribute is a boolean attribute.

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

Example

A form with a novalidate attribute:

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

HTML Form Elements

This chapter describes all the different HTML form elements.

The HTML <form> Elements

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

- <input>

- <label>
- <select>
- <textarea>
- <button>
- <fieldset>
- <legend>
- <datalist>
- <output>
- <option>
- <optgroup>

The <input> Element

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

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

Example

<label for="fname">First name:</label>
<input type="text" id="fname" name="fname">
All the different values of the type attribute are covered in the next chapter: HTML Input Types.

The <label> Element

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

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

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

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

The <select> Element

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

Example

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

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

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

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

Example

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

Visible Values:

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

Example

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

Allow Multiple Selections:

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

Example

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

The <textarea> Element

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

Example

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

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

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

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

The cat was playing in the garden.
You can also define the size of the text area by using CSS:

Example

```
<textarea name="message" style="width:200px; height:600px;">
The cat was playing in the garden.
</textarea>
```

The <button> Element

The <button> element defines a clickable button:

Example

<button type="button" onclick="alert(' World!')">Click Me!</button>
This is how the HTML code above will be displayed in a browser:

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

The <fieldset> and <legend> Elements

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

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

Example

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

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

Personalia:
First name:
John

Last name:
Doe

The <datalist> Element

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

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

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

Example

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

The <output> Element

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

Example

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

```
<form action="/action_page.php"
  oninput="x.value=parseInt(a.value)+parseInt(b.value)">
  0
```

```
  <input type="range"  id="a" name="a" value="50">
  100 +
  <input type="number" id="b" name="b" value="50">
  =
  <output name="x" for="a b"></output>
  <br><br>
  <input type="submit">
</form>
```

HTML Form Elements

Tag	Description
<form>	Defines an HTML form for user input
<input>	Defines an input control
<textarea>	Defines a multiline input control (text area)
<label>	Defines a label for an <input> element
<fieldset>	Groups related elements in a form
<legend>	Defines a caption for a <fieldset> element
<select>	Defines a drop-down list
<optgroup>	Defines a group of related options in a drop-down list
<option>	Defines an option in a drop-down list
<button>	Defines a clickable button
<datalist>	Specifies a list of pre-defined options for input controls
<output>	Defines the result of a calculation

HTML Input Types

This chapter describes the different types for the HTML <input>

element.

HTML Input Types

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

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

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

Input Type Text

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

Example

<form>

```
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```
This is how the HTML code above will be displayed in a browser:

First name:

Last name:

Input Type Password

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

Example

```
<form>
  <label for="username">Username:</label><br>
  <input type="text" id="username" name="username"><br>
  <label for="pwd">Password:</label><br>
  <input type="password" id="pwd" name="pwd">
</form>
```
This is how the HTML code above will be displayed in a browser:

Username:

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

Input Type Submit

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

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

input data.

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

Example

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

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

First name:
John
Last name:
Doe

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

Example

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

Input Type Reset

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

Example

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

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

First name:
John
Last name:
Doe

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

Input Type Radio

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

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

Example

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

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

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

HTML
CSS
JavaScript

Input Type Checkbox

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

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

Example

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

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

I have a bike
I have a car
I have a boat

Input Type Button

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

Example

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

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

Input Type Color

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

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

Example

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

Input Type Date

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

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

Example

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

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

Example

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

Input Type Datetime-local

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

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

Example

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

Input Type Email

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

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

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

Example

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

Input Type Image

The <input type="image"> defines an image as a submit button.

The path to the image is specified in the src attribute.

Example

```
<form>
<input type="image" src="img_submit.gif" alt="Submit" width="48" height="48">
</form>
```

Input Type File

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

Example

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

Input Type Hidden

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

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

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

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

Example

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

```
  <input type="hidden" id="custId" name="custId" value="3487">
  <input type="submit" value="Submit">
</form>
```

Input Type Month

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

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

Example

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

Input Type Number

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

You can also set restrictions on what numbers are accepted.

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

Example

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

Input Restrictions

Here is a list of some common input restrictions:

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

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

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

Example

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

Input Type Range

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

Example

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

Input Type Search

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

Example

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

Input Type Tel

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

Example

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

Input Type Time

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

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

Example

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

Input Type Url

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

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

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

Example

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

Input Type Week

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

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

Example

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

HTML Input Attributes

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

The value Attribute

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

Example

Input fields with initial (default) values:

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

The readonly Attribute

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

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

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

Example

A read-only input field:

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

The disabled Attribute

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

A disabled input field is unusable and un-clickable.

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

Example

A disabled input field:

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

The size Attribute

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

The default value for size is 20.

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

Example

Set a width for an input field:

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

The maxlength Attribute

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

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

Example

Set a maximum length for an input field:

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

The min and max Attributes

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

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

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

Example

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

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

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

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

The multiple Attribute

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

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

Example

A file upload field that accepts multiple values:

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

The pattern Attribute

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

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

Tip: Use the global title attribute to describe the pattern to help the user.

Tip: Learn more about regular expressions in our JavaScript tutorial.

Example

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

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

The placeholder Attribute

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

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

The placeholder attribute works with the following input types: text,

search, url, number, tel, email, and password.

Example

An input field with a placeholder text:

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

The required Attribute

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

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

Example

A required input field:

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

The step Attribute

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

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

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

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

Example

An input field with a specified legal number intervals:

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

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

The autofocus Attribute

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

Example

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

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

The height and width Attributes

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

Tip: Always specify both the height and width attributes for images. If height and width are set, the space required for the image is reserved when

the page is loaded. Without these attributes, the browser does not know the size of the image, and cannot reserve the appropriate space to it. The effect will be that the page layout will change during loading (while the images load).

Example

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

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

The list Attribute

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

Example

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

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

The autocomplete Attribute

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

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

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

Example

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

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

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

e Exercise

HTML Form and Input Elements

Tag	Description
<form>	Defines an HTML form for user input
<input>	Defines an input control

HTML Input form* Attributes

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

The form Attribute

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

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

Example

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

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

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

The formaction Attribute

The input formaction attribute specifies the URL of the file that will

process the input when the form is submitted.

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

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

Example

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

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

The formenctype Attribute

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

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

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

Example

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

```
<form action="/action_page_binary.asp" method="post">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
```

```
  <input type="submit" value="Submit">
  <input type="submit" formenctype="multipart/form-data"
  value="Submit as Multipart/form-data">
</form>
```

The formmethod Attribute

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

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

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

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

Notes on the "get" method:

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

Notes on the "post" method:

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

Example

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

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

The formtarget Attribute

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

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

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

Example

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

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

The formnovalidate Attribute

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

Note: This attribute overrides the novalidate attribute of the <form>

element.

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

Example

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

```
<form action="/action_page.php">
  <label for="email">Enter your email:</label>
  <input type="email" id="email" name="email"><br><br>
  <input type="submit" value="Submit">
  <input type="submit" formnovalidate="formnovalidate"
  value="Submit without validation">
</form>
```

The novalidate Attribute

The novalidate attribute is a <form> attribute.

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

Example

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

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

HTML Form and Input Elements

Tag	Description

<form>	Defines an HTML form for user input
<input>	Defines an input control

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

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:

Your browser does not support HTML5 video.

The HTML <video> Element

To show a video in HTML, use the <video> element:

Example

```
<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>
```

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

```
<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>
```

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

```
<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>
```

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	MediaType
MP4	video/mp4
WebM	video/webm
Ogg	video/ogg

L Audio/Video DOM Reference.

HTML Video Tags

Tag	Description
<video>	Defines a video or movie
<source>	Defines multiple media resources for media elements, such as <video> and <audio>
<track>	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

```
<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>
```

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

```
<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>
```

Note: Chromium browsers do not allow autoplay in most cases. However, muted autoplay is always allowed.

Add `muted` after `autoplay` to let your audio file start playing automatically (but muted):

Example

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

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	MediaType
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.

For a full DOM reference, go to our HTML Audio/Video DOM Reference.

HTML Audio Tags

Tag	Description
<audio>	Defines sound content
<source>	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

```
<object width="100%" height="500px"
data="snippet.html"></object>
```

Or images if you like:

Example

```
<object data="audi.jpeg"></object>
```

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

```
<embed src="audi.jpeg">
```

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

```
<embed width="100%" height="500px"
src="snippet.html">
```

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

```
<iframe width="420" height="315"
src="https://www.youtube.com/embed/tgbNymZ7vqY">
</iframe>
```

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

```
<iframe width="420" height="315"
src="https://www.youtube.com/embed/tgbNymZ7vqY?
autoplay=1&mute=1">
</iframe>
```

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

```
<iframe width="420" height="315"
src="https://www.youtube.com/embed/tgbNymZ7vqY?
playlist=tgbNymZ7vqY&loop=1">
</iframe>
```

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>
```

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.

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(showPosi
```

```
tion);
  } 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
- 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:

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)

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

Drag the hello 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(dat
a));
}
</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(dat
a));
}
```

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.

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.";
```