# Introduction to HTML

## What Is HTML?

HTML stands for *Hypertext Markup Language* and is most widely used to write web pages.

* 'Hypertext' refers to the way in which web pages (HTML documents) are linked together. Thus, the link available on a webpage is called Hypertext.
* 'Markup Language' means that you use HTML to simply "mark-up" a text document with tags that tell a web browser how to structure it to display.

Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, etc. to facilitate the sharing of scientific information between researchers.

Now HTML is being widely used to format web pages with the help of different tags available in HTML language.

## HTML Document Structure

A typical HTML document has the following structure:

<!-- This is document declaration tag -->

<!DOCTYPE html>

<html>

   <head>

      <!-- Document header related tags -->

   </head>

   <body>

      <!-- Document body related tags -->

   </body>

</html>

We will study all the header and body tags in subsequent chapters.

# HTML Components

## Tags

HTML tags are enclosed within angle braces <tag-name>. Except few tags, most of the tags have their corresponding closing tags </tag-name>. For example, <html> has its closing tag </html> and <body> tag has its closing tag </body> tag, etc.

The list below shows basic tags commonly used in HTML documents:

|  |  |  |
| --- | --- | --- |
| **Groups** | **Tags** | **Descriptions** |
| **Main structure** | <!DOCTYPE ...> | Used by web browsers to understand the version of the HTML used in the document.  Current version of HTML is 5 and it makes use of the following declaration: <!DOCTYPE html> |
| <html> | Encloses the complete HTML document and mainly comprises of:   * Document header (by <head>...</head> tags) * Document body (by <body>...</body> tags) |
| <head> | Represents the document's header which can keep other HTML tags like <title>, <link>, <meta>, <base>, <style>, <script>, and <noscript>.  For more details about these tags, check [this section](https://www.tutorialspoint.com/html/html_header.htm). |
| <body> | Represents the document's body which keeps other HTML tags like <h1>, <div>, <p>, etc. |
| <!-- ... --> | Represents a comment which is ignored by all web browsers. |
| **Metadata** | <title> | * Used inside the <head> tag. * Describe the document title. |
| <link> | * Used inside the <head> tag. * Define the relationship between the current document and an external resource. |
| **Structural** | <h1> to <h6> | * Used inside the <body> tag. * Represents a heading.   Note: You can use different sizes for your headings. HTML has 6 levels of headings, which use the elements <h1>, <h2>, <h3>, <h4>, <h5>, and <h6>. |
| <p> | * Used inside the <body> tag. * Represents a paragraph. |
| **Formatting**  **&**  **Layout** | <pre> | Forces text to follow the exact format of how it is written in the HTML document. |
| <b>, <i>, <u>, <strike>, <tt>, <sup>, <sub>, <ins>, <del>, <small>  <em>, <mark>, strong>, <abbr>, <acronym>, <q>, etc. | <https://www.tutorialspoint.com/html/html_formatting.htm>  <https://www.tutorialspoint.com/html/html_phrase_elements.htm> |
| <br /> | Breaks the current line and starts from the next line. |
| <hr /> | Creates a line from the current position in the document to the right margin and breaks the line accordingly. |
| <div> | Specifies a division or a section in a document. |
| **Image** | <img> | <https://www.tutorialspoint.com/html/html_images.htm> |
| **Table** | <table>  <tr>, <td>, <th>, <caption> | Check the [Table section](#_Table). |
| **List** | <ul>, <ol>, <li>, <dl>, <dd> | Check the [List section](#_List). |
| **Link** | <a> | <https://www.tutorialspoint.com/html/html_text_links.htm>  <https://www.tutorialspoint.com/html/html_image_links.htm>  <https://www.tutorialspoint.com/html/html_email_links.htm>  <https://www.tutorialrepublic.com/html-tutorial/html-links.php> |
| **Frame** | <frame>  <iframe> | <https://www.tutorialspoint.com/html/html_frames.htm>  <https://www.tutorialspoint.com/html/html_iframes.htm> |
| [**Grouping**](#_Grouping_Elements) | <div>  <span> | Check the [Block section](#_1fob9te). |
| **Semantics** | <article>  <aside>  <details>  <figcaption>  <figure>  <footer>  <header>  <main>  <mark>  <nav>  <section>  <summary>  <time> | Check the [Semantic Elements section](#_Semantic_Elements). |
| **Font**  **(face, size and** [**color**](#_Colors)**)** | <basefont>  <font> | <https://www.tutorialspoint.com/html/html_fonts.htm>  Note: The font tag deprecated and it is supposed to be removed in a future version of HTML. It's suggested to use CSS styles to manipulate your fonts. |
| [**Forms**](#_Forms) | <form> | <https://www.tutorialspoint.com/html/html_forms.htm> |
| **Embed multimedia** | <embed> | <https://www.tutorialspoint.com/html/html_embed_multimedia.htm> |
| **Marquees** | <marquee> | <https://www.tutorialspoint.com/html/html_marquees.htm> |
| [**CSS**](#_CSS) | <style> | Check [CSS section](#_2et92p0). |
| **Scripting** | <script>  <noscript> | Check JavaScript section. |
| **…** |  |  |

**Check a** [**full list of HTML5 tags**](https://www.tutorialrepublic.com/html-reference/html5-tags.php)**.**

## Elements

An HTML element is defined within **a start tag (or opening tag) and an end tag (or closing tag), with content in between.** For examples:

|  |  |  |
| --- | --- | --- |
| **Start Tag** | **Content** | **End Tag** |
| <p> | This is paragraph content. | </p> |
| <h1> | This is heading content. | </h1> |
| <br /> |  |  |

Here <p>....</p> is an HTML element, <h1>...</h1> is another HTML element.

**Note**

Some elements don't require the end tag to be present, simply because they don't need to be closed, such as <br />, <hr /> and <img ... />. These are known as **void elements**.

## Attributes

So far, we have seen some HTML tags and their usage in the simplest form. But most HTML tags can also have *attributes*, providing extra bits of information.

An attribute is used to define the **characteristics of an HTML element** and is **placed inside the element's opening tag**. All attributes are made up of two parts:

* The *name* is the property you want to set. For example, the paragraph <p> element in the example carries an attribute whose name is align, which you can use to indicate the alignment of paragraph on the page.
* The *value* is what you want the value of the property to be set and always put within quotations.

The below example shows 3 possible values of align attribute: left, center and right.

<!DOCTYPE html>

<html>

   <head>

      <title>Align attribute example</title>

   </head>

   <body>

      <p align = "left">This is left aligned</p>

      <p align = "center">This is center aligned</p>

      <p align = "right">This is right aligned</p>

   </body>

</html>

Output:

This is left aligned

This is center aligned

This is right aligned

**Notes**:

* Attribute names and values are **case-insensitive**. However, the World Wide Web Consortium (W3C) recommends lowercase formatting.
* Both **single and double quotes** can be used to quote attribute values. However, double quotes are most common. In situations where the attribute value itself contains double quotes, it is necessary to wrap the value in single quotes, e.g., value='John "Williams" Jr.'
* **Some attributes in HTML5 don't consist of name/value pairs but consists of just name**. These are called *Boolean* attributes. For examples: checked, disabled, readonly, required, etc. Check [common Boolean attributes](https://www.tutorialrepublic.com/codelab.php?topic=html&file=boolean-attributes).

### Core Attributes

The four core attributes that can be used on the majority of HTML elements are:

#### The 'id' Attribute

It uniquely identifies any element within an HTML page. There are two primary reasons that you might want to use it on an element:

* If you have two elements of the same name within a web page (or style sheet), you can use two different id attributes to distinguish between elements.
* If an element carries an id attribute as a unique identifier, it is possible to identify just that element and its content.

Example:

<p id = "html">This explains what is HTML</p>

<p id = "css">This explains what is Cascading Style Sheet</p>

#### The 'title' Attribute

It gives a suggested title for the element. The behavior of this attribute will depend upon the element that carries it, although it is often displayed as a tooltip when cursor comes over the element or while the element is loading.

Example:

<!DOCTYPE html>

<html>

<head>

<title>The title attribute example</title>

</head>

<body>

<h1 title = "Hello HTML!">Titled heading tag example</h3>

</body>

</html>

Output:

**Titled heading tag example**

Now try to bring your cursor over "Titled heading tag example" and you will see "Hello HTML!" is coming out as a tooltip of the cursor.

#### The 'class' Attribute

It associates an element with CSS, and specifies the class of element. You will learn more about the use of the class attribute when you will learn [Cascading Style Sheet (CSS)](#_2et92p0). So for now you can avoid it.

The value of the attribute may also be a space-separated list of class names. For example:

class = "class-name1 class-name2 class-name3"

#### The 'style' Attribute

It allows you to specify CSS rules within the element.

For example:

<!DOCTYPE html>

<html>

<head>

<title>The style attribute</title>

</head>

<body>

<p style = "font-family:arial; color:#FF0000;">Some text...</p>

</body>

</html>

Output:

Some text...

### Internationalization Attributes

The three internationalization attributes that can be used on the majority of HTML elements are:

#### The 'dir' Attribute

It represents the direction in which the text should flow. It can take one of two values, as you can see in the table:

|  |  |
| --- | --- |
| **Value** | **Meaning** |
| ltr | Left to right (the default value) |
| rtl | Right to left (for languages such as Hebrew or Arabic that are read right to left) |

Example:

<!DOCTYPE html>

<html dir = "rtl">

<head>

<title>Display directions</title>

</head>

<body>

This is how IE 5 renders right-to-left directed text.

</body>

</html>

Output:

This is how IE 5 renders right-to-left directed text.

#### The 'lang' Attribute

It indicates the main language used in a document, but this attribute was kept in HTML only for backwards compatibility with earlier versions of HTML. This attribute has been replaced by the xml:lang attribute in new XHTML documents.

The values of the lang attribute are ISO-639 standard two-character language codes. Check [HTML Language Codes: ISO 639](https://www.tutorialspoint.com/html/language_iso_codes.htm) for a complete list of language codes.

Example:

<!DOCTYPE html>

<html lang = "en">

<head>

<title>English Language Page</title>

</head>

<body>

This page is using English Language

</body>

</html>

Output:

This page is using English Language

# Grouping Elements

## Different Types of Elements

All HTML elements can be categorized into two categories (a) Block-Level Elements (b) Inline Elements.

### Block Elements

Block elements all **start on their own new line**, and anything that follows them appears on its own new line. For example: <p>, <h1>, <h2>, <h3>, <h4>, <h5>, <h6>, <ul>, <ol>, <dl>, <pre>, <hr />, <blockquote>, <address>, etc.

### Inline Elements

Inline elements, on the other hand, can **appear within sentences** and do not have to appear on a new line of their own. For example: <span>, <b>, <i>, <u>, <em>, <strong>, <sup>, <sub>, <big>, <small>, <li>, <ins>, <del>, <code>, <cite>, <dfn>, <kbd>, <var>, etc.

## How to Group Elements

There are two important tags which we use very frequently to group various other HTML tags: (a) <div> and (b) <span>.

### The <div> Tag

This block-level tag plays a big role in grouping various HTML tags and applying CSS on group of elements. Moreover, it's also the most common way to define [HTML layouts](#_tyjcwt).

<!DOCTYPE html>

<html>

   <head>

      <title>HTML div Tag</title>

   </head>

   <body>

      <!-- First group of tags -->

      <div style = "color:red">

         <p>Following is a list of vegetables:</p>

         <ul>

            <li>Beetroot</li>

            <li>Ginger</li>

         </ul>

      </div>

      <!-- Second group of tags -->

      <div style = "color:green">

         <p>Following is a list of fruits:</p>

         <ul>

            <li>Apple</li>

            <li>Banana</li>

         </ul>

      </div>

   </body>

</html>

Output:

|  |
| --- |
| Following is a list of vegetables:   * Beetroot * Ginger   Following is a list of fruits:   * Apple * Banana |

### The <span> Tag

This inline tag is used to group inline-elements in an HTML document. It is mostly used with CSS, like the <div> tag.

<!DOCTYPE html>

<html>

   <head>

      <title>HTML span Tag</title>

   </head>

   <body>

      <p>This is <span style = "color:red">red</span> and this is

         <span style = "color:green">green</span>

</p>

   </body>

</html>

Output:

This is red and this is green

# Colors

The <body> tag has following attributes which can be used to set different colors:

* bgcolor − sets a color for the background of the page.
* text − sets a color for the body text.
* link − sets a color for linked text.
* alink − sets a color for active links or selected links.
* vlink − sets a color for visited links − that is, for linked text that you have already clicked on.

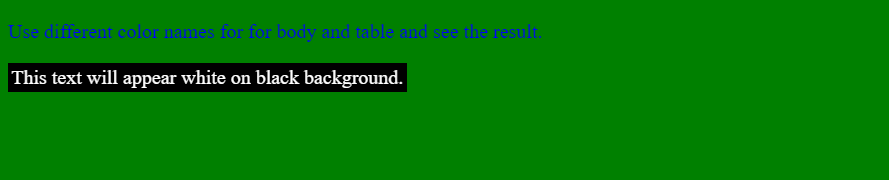
## HTML Color Coding Methods

There are following three different methods to set colors in your web page:

* Color names − You can specify color names directly like green, blue or red.
* Hex codes − A six-digit code representing the amount of red, green, and blue that makes up the color.
* Color decimal or percentage values − This value is specified using the rgb( ) property.

**Note**: Check a complete list of [HTML Color Name.](https://www.tutorialspoint.com/html/html_color_names.htm)

Below examples produce the same output as follows:



### Color Names

<!DOCTYPE html>

<html>

   <head>

      <title>HTML Colors by Name</title>

   </head>

   <body text = "blue" bgcolor = "green">

      <p>Use different color names for body and table and see the result.</p>

      <table bgcolor = "black">

         <tr>

            <td>

               <font color = "white">This text appears white on black bgr.</font>

            </td>

         </tr>

      </table>

   </body>

</html>

### Hex Codes

<!DOCTYPE html>

<html>

   <head>

      <title>HTML Colors by Hex</title>

   </head>

   <body text = "#0000FF" bgcolor = "#00FF00">

      <p>Use different color hexa for body and table and see the result.</p>

      <table bgcolor = "#000000">

         <tr>

            <td>

               <font color = "#FFFFFF">This text appears white on black bgr.</font>

            </td>

         </tr>

      </table>

   </body>

</html>

### RGB Values

<!DOCTYPE html>

<html>

   <head>

      <title>HTML Colors by RGB code</title>

   </head>

   <body text = "rgb(0,0,255)" bgcolor = "rgb(0,255,0)">

      <p>Use different color code for body and table and see the result.</p>

      <table bgcolor = "rgb(0,0,0)">

         <tr>

            <td>

               <font color = "rgb(255,255,255)">This text appears white on black bgr.</font>

            </td>

         </tr>

      </table>

   </body>

</html>

# CSS

## What Is CSS?

Cascading Style Sheets (CSS) describe **how documents are presented** on screens. It provides easy and effective alternatives to specify various attributes for the HTML tags. So you can easily specify a number of style properties for a given HTML element.

Each CSS property has the form:

property-name:property-value;

## Non-CSS vs. CSS

Consider the following example which makes use of <font> tag and associated attributes to specify text color and font size:

<!DOCTYPE html>

<html>

   <head>

      <title>HTML CSS</title>

   </head>

   <body>

      <p><font color = "green" size = "5">Hello, World!</font></p>

   </body>

</html>

We can re-write above example with the help of CSS as follows:

<!DOCTYPE html>

<html>

   <head>

      <title>HTML CSS</title>

   </head>

   <body>

      <p style = "color:green; font-size:24px;" >Hello, World!</p>

   </body>

</html>

Both code give the same output:

Hello, World!

## Different Ways to Use CSS in HTML

### External Style Sheet

If you need to use style sheet rules to various pages, then it's always recommended to define a common style sheet in a separate file. A cascading style sheet file will have extension as .css and it will be included in HTML files using <link> tag.

Consider a style sheet file named style.css which has three CSS rules applicable to three classes defined for the HTML tags.

.red {

    color: red;

 }

 .green {

    color:green;

 }

 .thick {

    font-size:20px;

 }

Now let's make use of the above external CSS file in our HTML document:

<!DOCTYPE html>

<html>

   <head>

      <title>HTML External CSS</title>

      <link rel = "stylesheet" type = "text/css" href = "/html/style.css">

   </head>

   <body>

      <p class = "red">This is red</p>

      <p class = "thick">This is thick</p>

      <p class = "green">This is green</p>

      <p class = "thick green">This is thick and green</p>

   </body>

</html>

Output:

This is red

This is thick

This is green

This is thick and green

### Internal Style Sheet

If you want to apply style sheet rules to a single document only, then you can include those rules in header section of the HTML document using <style> tag.

**Note**: Rules defined in internal style sheet overrides the rules defined in an external CSS file.

Let's re-write above example:

<!DOCTYPE html>

<html>

   <head>

      <title>HTML Internal CSS</title>

      <style type = "text/css">

         .red {

            color: red;

         }

         .green {

            color:green;

         }

         .thick{

            font-size:20px;

         }

      </style>

   </head>

   <body>

      <p class = "red">This is red</p>

      <p class = "thick">This is thick</p>

      <p class = "green">This is green</p>

      <p class = "thick green">This is thick and green</p>

   </body>

</html>

### Inline Style Sheet

You can apply style sheet rules directly to any HTML element using style attribute of the relevant tag. This should be done only when you are interested to make a particular change in any HTML element only.

**Note**: Rules defined inlinely with the element overrides both the rules defined in an external CSS file and in an internal style sheet.

Let's re-write above example:

<!DOCTYPE html>

<html>

   <head>

      <title>HTML Inline CSS</title>

   </head>

   <body>

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

      <p style = "font-size:20px;">This is thick</p>

      <p style = "color:green;">This is green</p>

      <p style = "color:green; font-size:20px;">This is thick and green</p>

   </body>

</html>

## List of CSS Properties

<https://www.w3schools.com/cssref/>

<https://cssreference.io/>

### Size

#### box-sizing

By default, the width and height of an element is calculated like this:

* width + padding + border = actual width of an element
* height + padding + border = actual height of an element

This means: When you set the width/height of an element, the element often appears bigger than you have set.

**Example**: The following shows two <div> elements with the same specified width and height, but they end up with different sizes in the result:

.div1 {

    width: 300px;

    height: 100px;

    border: 1px solid blue;

}

.div2 {

    width: 300px;

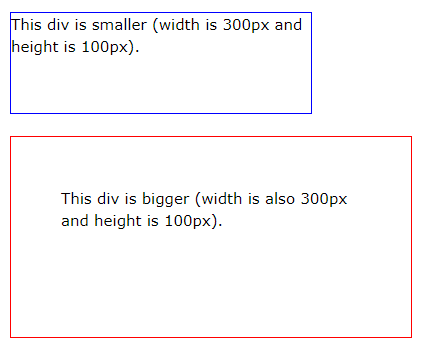
    height: 100px;

    padding: 50px;

    border: 1px solid red;

}

Output:



The box-sizing property solves this problem by **including the padding and border in an element's total width and height**.

**Example**: Both divs are the same size now!

.div1 {

    width: 300px;

    height: 100px;

    border: 1px solid blue;

    box-sizing: border-box;

}

.div2 {

    width: 300px;

    height: 100px;

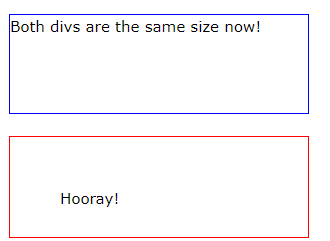
    padding: 50px;

    border: 1px solid red;

    box-sizing: border-box;

}

Output:



**Tip:**

Since the result of using the box-sizing: border-box; is so much better, many developers want all elements on their pages to work this way. The code below ensures that all elements are sized in this more intuitive way.

\* {

    box-sizing: border-box;

}

### Position

**Usage**: Specifies the type of positioning method used for an element (relative, absolute, etc.).

**Syntax**: position: static|absolute|fixed|relative|sticky|initial|inherit

**Values**:

<https://www.w3schools.com/cssref/pr_class_position.asp>

**Note**: Properties top, bottom, left, right must be used with the position property.

## CSS Selectors

Selectors are used to **"find" (or select) the HTML elements you want to style**.

We can divide CSS selectors into 5 categories:

* Simple selectors (select elements based on name, id, class)
* [Combinator selectors](https://www.w3schools.com/css/css_combinators.asp) (select elements based on a specific relationship between them)
* [Pseudo-class selectors](https://www.w3schools.com/css/css_pseudo_classes.asp) (select elements based on a certain state)

* [Pseudo-elements selectors](https://www.w3schools.com/css/css_pseudo_elements.asp) (select and style a part of an element)
* [Attribute selectors](https://www.w3schools.com/css/css_attribute_selectors.asp) (select elements based on an attribute or attribute value)

### Simple Selectors

#### Element Selector

Selects HTML elements based on their name.

Syntax: Name the class the same as the HTML element you want to style.

**Example**: All <p> elements on the page will be center-aligned, with a red text color:

<!DOCTYPE html>

<html>

<head>

<style>

    p {     /\* It must be named as 'p', not 'p1' or 'p2' \*/

        text-align: center;

        color: red;

    }

</style>

</head>

<body>

<p>Every paragraph will be affected by the style.</p>

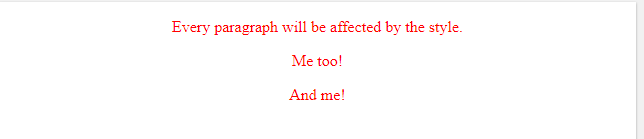
<p id="para1">Me too!</p>

<p>And me!</p>

</body>

</html>

Output:



#### The 'id' Selector

Uses the id attribute of an HTML element to select a specific element.

Note: The id of an element is unique within a page.

Syntax: Use a hash (#) character, followed by the id of the element.

**Example**: The style is applied to the HTML element with id="para1":

<!DOCTYPE html>

<html>

<head>

    <style>

        #para1 {

            text-align: center;

            color: red;

        }

    </style>

</head>

<body>

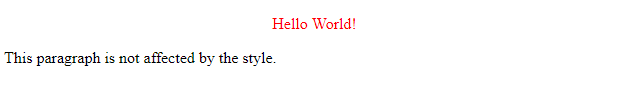
    <p id="para1">Hello World!</p>

    <p>This paragraph is not affected by the style.</p>

</body>

</html>

Output:



#### The 'class' Selector

Selects HTML elements based on their class attribute.

Syntax: Use a period (.) character, followed by the class name.

**Example 1**: Only HTML element with class="center" will be red and center-aligned:

<!DOCTYPE html>

<html>

<head>

    <style>

        .center {

            text-align: center;

            color: red;

        }

    </style>

</head>

<body>

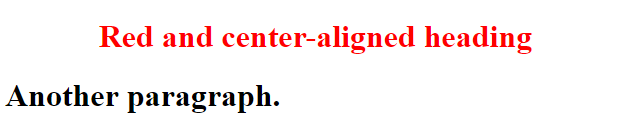
    <h1 class="center">Red and center-aligned heading</h1>

    <h1>Another paragraph.</h1>

</body>

</html>

Output:



**Example 2**: You can specify that only specific HTML elements should be affected by a class.

<!DOCTYPE html>

<html>

<head>

    <style>

        p.center {

            text-align: center;

            color: red;

        }

    </style>

</head>

<body>

    <h1 class="center">This heading will not be affected</h1>

    <p class="center">This paragraph will be red and center-aligned.</p>

</body>

</html>

Output:



HTML elements can also refer to more than one class.

#### Universal Selector

Selects all HTML elements on the page.

Syntax: Replace class name by a star (\*) character.

**Example**: This will affect every HTML element on the page:

<!DOCTYPE html>

<html>

<head>

    <style>

        \* {

            text-align: center;

            color: blue;

        }

    </style>

</head>

<body>

    <h1>Hello world!</h1>

    <p>Every element on the page will be affected by the style.</p>

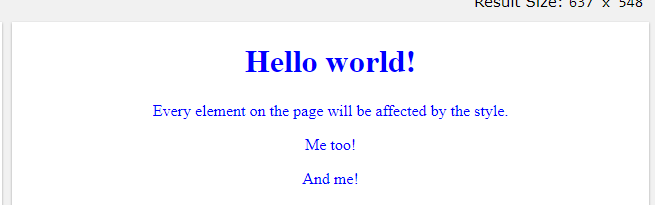
    <p id="para1">Me too!</p>

    <p>And me!</p>

</body>

</html>

Output:



#### Grouping Selector

Selects all the HTML elements with the same style definitions.

**Example**:

Look at the following CSS code (the h1 and h2 elements have the same style definitions):

/\* Lengthy way \*/

h1 {

  text-align: center;

  color: red;

}

h2 {

  text-align: center;

  color: red;

}

It will be better to group the selectors, to minimize the code.

Syntax: To group selectors, separate each selector with a comma (',').

/\* Better way \*/

h1, h2 {

  text-align: center;

  color: red;

}

### Combinator Selectors

#### Descendant Selector (Space)

**Example**: "div p" select all <p> elements that are inside the <div> element.

<!DOCTYPE html>

<html>

<head>

    <style>

        div p {

            background-color: yellow;

        }

    </style>

</head>

<body>

    <h2>Descendant Selector</h2>

    <p>The descendant selector matches all elements that are descendants of a specified element.</p>

    <div>

        <p>Paragraph 1 in the div.</p>

        <p>Paragraph 2 in the div.</p>

        <section>

            <p>Paragraph 3 in the div.</p>

        </section>

    </div>

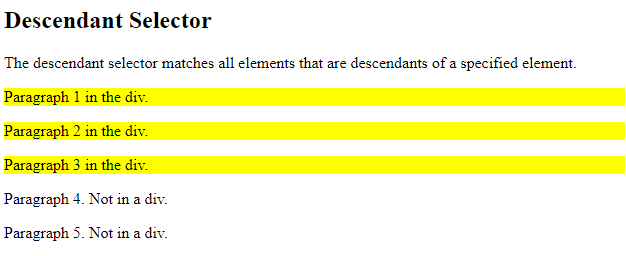
    <p>Paragraph 4. Not in a div.</p>

    <p>Paragraph 5. Not in a div.</p>

</body>

</html>

Output:



#### Child Selector (>)

**Example**: "div > p" selects all <p> elements that are the children of the <div> element.

<!DOCTYPE html>

<html>

<head>

    <style>

        div > p {

            background-color: yellow;

        }

    </style>

</head>

<body>

    <h2>Child Selector</h2>

    <p>The child selector (>) selects all elements that are the children of a specified element.</p>

    <div>

        <p>Paragraph 1 in the div.</p>

        <p>Paragraph 2 in the div.</p>

        <section>

            <p>Paragraph 3 in the div.</p> <!-- not Child but Descendant -->

        </section>

        <p>Paragraph 4 in the div.</p>

    </div>

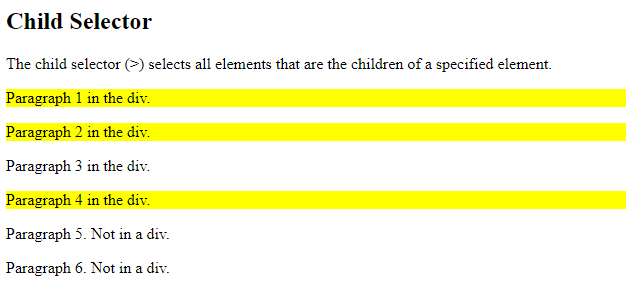
    <p>Paragraph 5. Not in a div.</p>

    <p>Paragraph 6. Not in a div.</p>

</body>

</html>

Output:



#### Adjacent Sibling Selector (+)

**Example**: "div + p" select the first <p> element that are placed immediately after <div> elements.

<!DOCTYPE html>

<html>

<head>

    <style>

        div + p {

            background-color: yellow;

        }

    </style>

</head>

<body>

    <h2>Adjacent Sibling Selector</h2>

    <p>The + selector selects an element that is directly after another specific element.</p>

    <p>The following example selects the first p element placed immediately after div elements:</p>

    <div>

        <p>Paragraph 1 in the div.</p>

        <p>Paragraph 2 in the div.</p>

    </div>

    <p>Paragraph 3. After a div.</p>

    <p>Paragraph 4. After a div.</p>

    <div>

        <p>Paragraph 5 in the div.</p>

        <p>Paragraph 6 in the div.</p>

    </div>

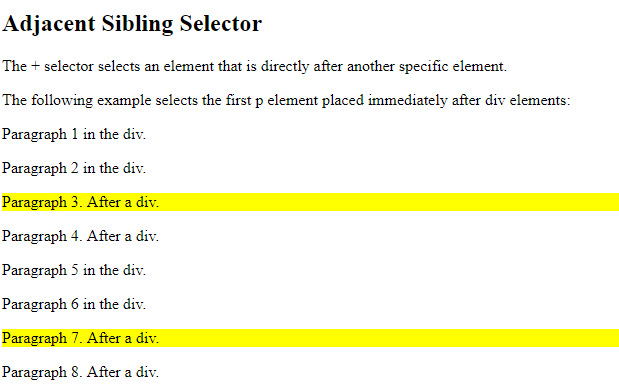
    <p>Paragraph 7. After a div.</p>

    <p>Paragraph 8. After a div.</p>

</body>

</html>

Output:



#### General Sibling Selector (~)

**Example**: "div ~ p" select every <p> element that are preceded by a <div> element

<!DOCTYPE html>

<html>

<head>

    <style>

        div ~ p {

            background-color: yellow;

        }

    </style>

</head>

<body>

    <h2>General Sibling Selector</h2>

    <p>The general sibling selector (~) selects all elements that are siblings of a specified element.</p>

    <p>Paragraph 1.</p>

    <div>

        <p>Paragraph 2.</p>

    </div>

    <p>Paragraph 3.</p>

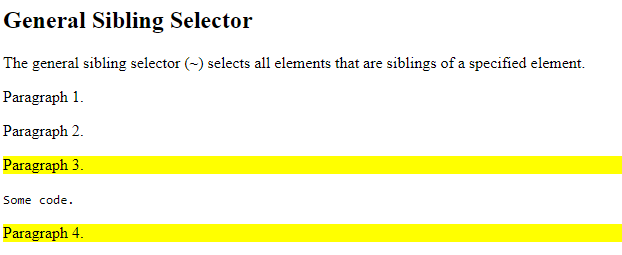
    <code>Some code.</code>

    <p>Paragraph 4.</p>

</body>

</html>

Output:



### CSS Pseudo-Classes

A pseudo-class is used to define a special state of an element.

For example, it can be used to:

* Style an element when a user mouses over it
* Style visited and unvisited links differently
* Style an element when it gets focus

**Example 1**:

<!DOCTYPE html>

<html>

<head>

    <style>

        /\* unvisited link \*/

        a:link {

            color: red;

        }

        /\* visited link \*/

        a:visited {

            color: green;

        }

        /\* mouse over link \*/

        a:hover {

            color: hotpink;

        }

        /\* selected link \*/

        a:active {

            color: blue;

        }

        /\* first child \*/

        p:first-child {

            color: blue;

        }

    </style>

</head>

<body>

    <p>This is some text.</p>

    <p>This is some text.</p>

    <h2>CSS Links</h2>

    <p><b><a href="default.asp" target="\_blank">This is a link</a></b></p>

</body>

</html>

Output:



**Example 2**: Tooltip

<!DOCTYPE html>

<html>

<head>

    <style>

        p {

            display: none;

            background-color: yellow;

            padding: 20px;

        }

        div:hover p {

            display: block;

        }

    </style>

</head>

<body>

    <div>Hover over me to show the p element

        <p>Tada! Here I am!</p>

    </div>

</body>

</html>

Output: Hover the mouse on the <div>, it will show <p>



### Pseudo-Elements Selectors

[Pseudo-elements selectors](https://www.w3schools.com/css/css_pseudo_elements.asp) (select and style a part of an element)

### Attribute Selectors

[Attribute selectors](https://www.w3schools.com/css/css_attribute_selectors.asp) (select elements based on an attribute or attribute value)

# Table

## HTML Element

The HTML tables are created using the <table> element with following sub elements:

* <tr>: Create table **rows**.
* <td>: Put inside the <tr> to create data **cells**. Text under <td> are regular and left-aligned by default
* <th>: Replace <td> to represent table **heading** for the first row of the table. Text under <th> are bold and centered-aligned by default.
* <caption>: Create a **title** or explanation for the table It shows up at the top of the table and centered-aligned by default.

**Example**:

<!DOCTYPE html>

<html>

   <head>

      <title>HTML Tables</title>

   </head>

   <body>

    <table border = "1">

        <caption>This is a caption</caption>

        <tr>

           <th>Name</th>

           <th>Salary</th>

        </tr>

        <tr>

           <td>Ramesh Raman</td>

           <td>5000</td>

        </tr>

        <tr>

           <td>Shabbir Hussein</td>

           <td>7000</td>

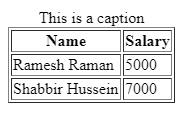
        </tr>

     </table>

   </body>

</html>

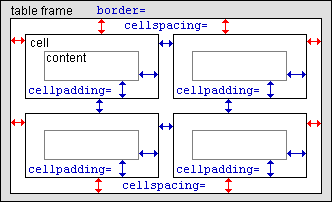
Output:



## HTML Attributes

**Padding:**

* cellpadding and cellspacing: Adjust the white space in your table cells. Difference between them:



**Merge cells:**

* colspan: Merge two or more columns into a single column.
* rowspan: Merge two or more rows into a single row.

**Color:**

* bgcolor: Set background color for whole table or just for one cell.
* background: Set background image for whole table or just for one cell.
* bordercolor: Set border color

Note: The bgcolor, background, and bordercolor attributes are deprecated in HTML5. DO NOT use them!

**Size:**

* width: Set width for table (cells in the table will be affected too)
* height: Set height for table (cells in the table will be affected too)

Tip: You can specify table width or height in terms of pixels, or percentage of available screen area.

**Alignment:**

* align: Set alignment (left, right or center) for text in table rows and cells

**Example**:

<!DOCTYPE html>

<html>

   <head>

      <title>HTML Tables</title>

   </head>

   <body>

    <table border = "1" cellpadding = "5" cellspacing = "0"

           bordercolor = "red" bgcolor = "lightblue"

           width = "300">

        <tr>

           <th>Name</th>

           <th>Salary</th>

        </tr>

        <tr>

           <td>Ramesh Raman</td>

           <td>5000</td>

        </tr>

        <tr>

           <td>Shabbir Hussein</td>

           <td>7000</td>

        </tr>

        <tr>

            <td colspan = 2>End of the table</td>

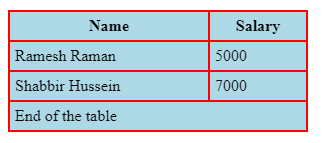
        </tr>

     </table>

   </body>

</html>

Output:



## CSS Properties

|  |  |  |  |
| --- | --- | --- | --- |
| **Properties** | **Description** | **Example** | **Counterpart in HTML attributes** |
| border | Set weight, style and color for border | table, th, td {  border: 1px solid black;  } | border = "1" |
| border-collapse | Let the borders collapse into one border. | table, th, td {  border-collapse: collapse;  } | cellspacing = "0" |
| padding | Set the padding for cells. | th, td {  padding: 15px;  } | cellpadding= "15" |
| border-spacing | Set space between cells.  Note: If the table has collapsed borders, border-spacing has no effect. | table {  border-spacing: 5px;  } | cellspacing = "5" |
| text-align | Align for text in rows. | th {  text-align: left;  } |  |
| background-color | Set background color for table. | table {  background-color: #f1f1c1;  } | bgcolor = #f1f1c1" |

**Example**:

<!DOCTYPE html>

<html>

<head>

    <title>HTML Tables</title>

    <style type="text/css">

        table, th, td {

            border-collapse: collapse;      /\* Similar to: cellspacing = "0" \*/

            border: 1px solid red;          /\* Similar to: border = "1" bordercolor = "red" \*/

            padding: 5px;                   /\* Similar to: cellpadding = "5" \*/

            text-align: center;

        }

        table {

            border-spacing: 15px;           /\* Not work if border-collapse: collapse; \*/

            background-color: lightblue;    /\* Similar to: bgcolor = "lightblue" \*/

        }

    </style>

</head>

<body>

    <table width="100%">

        <tr>

            <th>Name</th>

            <th>Salary</th>

        </tr>

        <tr>

            <td>Ramesh Raman</td>

            <td>5000</td>

        </tr>

        <tr>

            <td>Shabbir Hussein</td>

            <td>7000</td>

        </tr>

        <tr>

            <td colspan=2>End of the table</td>

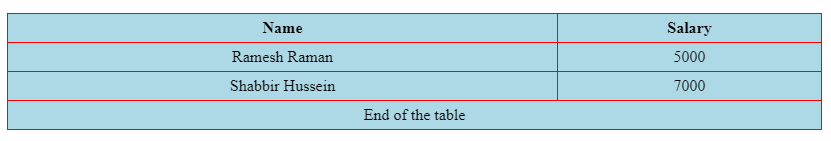
        </tr>

    </table>

</body>

</html>

Output:



## Others

### Special Style for One Table

Example:

<!DOCTYPE html>

<html>

<head>

    <style>

        table {

            width: 100%;

        }

        table, th, td {

            border: 1px solid black;

            border-collapse: collapse;

        }

        th, td {

            padding: 15px;

            text-align: left;

        }

        #t01 tr:nth-child(even) {

            background-color: blue;

        }

        #t01 tr:nth-child(odd) {

            background-color: green;

        }

        #t01 th {

            background-color: black;

            color: white;

        }

    </style>

</head>

<body>

    <h2>Styling Tables</h2>

    <table>

        <tr>

            <th>Firstname</th>

            <th>Lastname</th>

            <th>Age</th>

        </tr>

        <tr>

            <td>Jill</td>

            <td>Smith</td>

            <td>50</td>

        </tr>

        <tr>

            <td>Eve</td>

            <td>Jackson</td>

            <td>94</td>

        </tr>

    </table>

    <br>

    <table id="t01">

        <tr>

            <th>Firstname</th>

            <th>Lastname</th>

            <th>Age</th>

        </tr>

        <tr>

            <td>Jill</td>

            <td>Smith</td>

            <td>50</td>

        </tr>

        <tr>

            <td>Eve</td>

            <td>Jackson</td>

            <td>94</td>

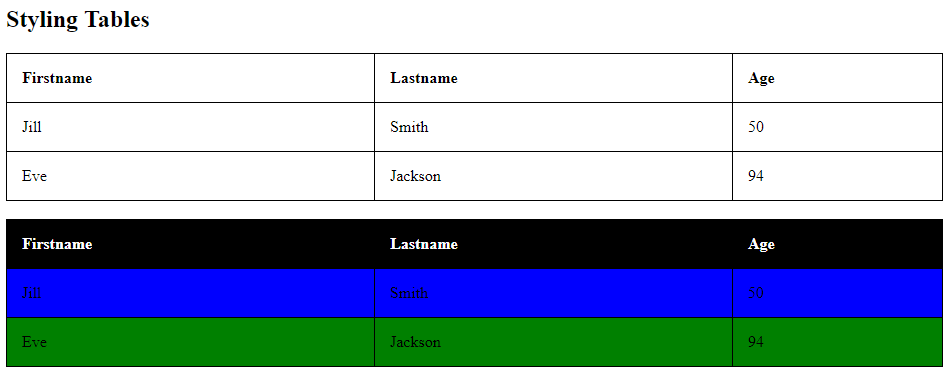
        </tr>

    </table>

</body>

</html>

Output:



# List

* An **ordered list** starts with a <ol> tag. Each list item starts with the <li> tag. To specify the type of numbering, use the type attribute. Following are the possible options:
  + <ol type = "1"> - Default
  + <ol type = "I">
  + <ol type = "i">
  + <ol type = "A">
  + <ol type = "a">
* An **unordered list** starts with a <ul> tag. Each list item starts with the <li> tag.
* A **description list** (or definition list) starts with a <dl> tag. The <dd> tag describes each term.

**Example:**

<!DOCTYPE html>

<html>

<body>

<h2>Ordered list</h2>

<h3>Number numerals</h3>

<ol>

  <li>Coffee</li>

  <li>Tea</li>

  <li>Milk</li>

</ol>

<h3>Uppercase numerals</h3>

<ol type = "I">

  <li>Coffee</li>

  <li>Tea</li>

  <li>Milk</li>

</ol>

<h3>Lowercase letters</h3>

<ol type = "a">

  <li>Coffee</li>

  <li>Tea</li>

  <li>Milk</li>

</ol>

<h2>Unordered list</h2>

<ul>

  <li>Coffee</li>

  <li>Tea</li>

  <li>Milk</li>

</ul>

<h2>Description list</h2>

<dl>

  <dt>Coffee</dt>

  <dd>- black hot drink</dd>

  <dt>Milk</dt>

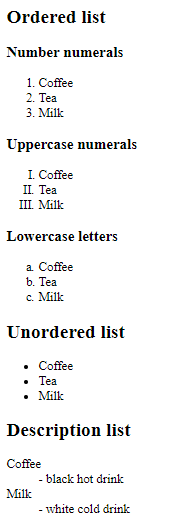
  <dd>- white cold drink</dd>

</dl>

</body>

</html>

Output:



# Forms

The <form> element is used to create an HTML form for user input. It's a container for different types of input elements, such as text fields, checkboxes, radio buttons, submit buttons, etc.

[HTML Form Attributes (w3schools.com)](https://www.w3schools.com/html/html_forms_attributes.asp)

Sub elements used in the <form> element can be:

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

## Types of Forms

### Input Types

#### Text Fields

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

**Example**:

<form>

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

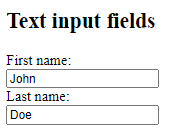
    <input type="text" id="fname" name="fname" value="John"><br>

    <label for="lname">Last name:</label><br>

    <input type="text" id="lname" name="lname" value="Doe">

</form>

**Output**:



#### Radio Buttons

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

**Example**:

<form>

  <input type="radio" id="male" name="gender" value="male">

  <label for="male">Male</label><br>

  <input type="radio" id="female" name="gender" value="female">

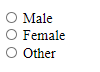
  <label for="female">Female</label><br>

  <input type="radio" id="other" name="gender" value="other">

  <label for="other">Other</label>

</form>

**Output**:



#### Checkboxes

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

**Example**:

<form action="/action\_page.php">

  <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">

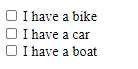
  <label for="vehicle2"> I have a car</label><br>

  <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat">

  <label for="vehicle3"> I have a boat</label><br><br>

</form>

**Output**:



#### 'Submit' Button

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

**Example**:

<form action="/action\_page.php">

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

  <input type="text" id="fname" name="fname" value="John"><br>

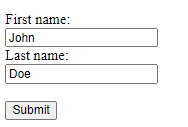
  <label for="lname">Last name:</label><br>

  <input type="text" id="lname" name="lname" value="Doe"><br><br>

  <input type="submit" value="Submit">

</form>

**Output**:



If you click the "Submit" button, the form-data will be sent to a page called "/action\_page.php".

#### Others

[HTML Input Types (w3schools.com)](https://www.w3schools.com/html/html_form_input_types.asp)

[HTML Input Attributes (w3schools.com)](https://www.w3schools.com/html/html_form_attributes.asp)

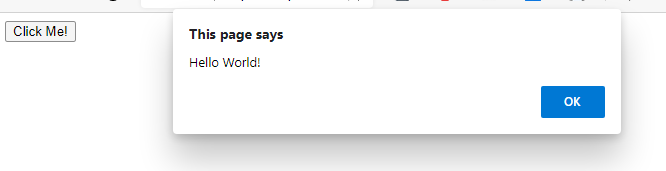
### Button

The <button> element defines a clickable button.

**Example 1**: Display alert box

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

**Output**:



**Example 2**: Redirect to another page

<button type="button" onclick="document.location.href='another/page'">Click Me!</button>

### Drop Down List

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

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> tag:

**Example**:

<form>

  <label for="cars">Choose a car:</label>

  <select id="cars" name="cars">

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

    <option value="volvo" selected>Volvo</option>

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

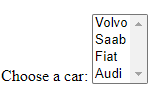
  </select>

</form>

**Output**:



Tip: To allow the user to select more than one value, add the multiple attribute to the <select> tag:



### Combo Box

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>

    <input list="browsers" name="browser">

    <datalist id="browsers">

      <option value="Internet Explorer">

      <option value="Firefox">

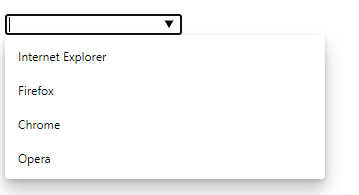
      <option value="Chrome">

      <option value="Opera">

    </datalist>

</form>

**Output**:



### Text Area

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

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

For example:

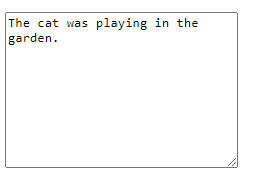
<form>

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

  <br><br>

</form>

Output:



### Group

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

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

**Example**:

<form>

    <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>

    </fieldset>

</form>

**Output:**



## HTML Elements

### The <input> Element

The <input> element is the most used form element. It can be displayed in many ways, depending on the type attribute. For 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 |

### The <label> Element

The <label> tag defines a label for many form elements. It's useful for:

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

## HTML Attributes

### The Name Attribute for <input>

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**:

<form action="/action\_page.php">

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

  <input type="text" id="fname" name="fname" value="John"><br><br>

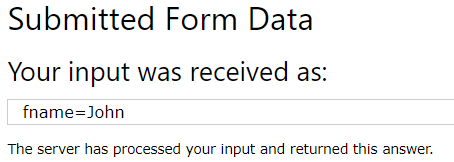
  <input type="submit" value="Submit">

</form>

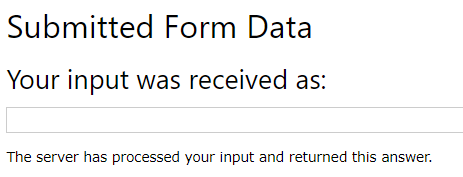
**Output**:



Assume code for action\_page.php is available. If you click the "Submit" button, the value of the input field will be sent:



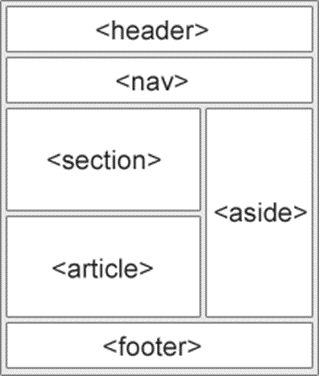
However, if you remove the name attribute in the input tag, the value of the input field will NOT be sent:



# Layouts

There are **various techniques** to create multicolumn layouts. Each technique has its pros and cons**.**

## HTML Layout Elements

HTML has several [semantic elements](#_Semantic_Elements) that define the different parts of a web page:

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

## Layout Techniques

### CSS Framework

If you want to create your layout **fast**, you can use a CSS framework, like [W3.CSS](https://www.w3schools.com/w3css/default.asp) or [Bootstrap](https://www.w3schools.com/bootstrap/default.asp).

### CSS Grid

Using the HTML <div> element in combination with the CSS Grid Layout Module is the most common method of creating layouts in HTML.

**Warning**

In addition to using <div> element for creating layouts in HTML, we can use **table** (which is actually the simplest way), but it's NOT recommended. Layouts created using tables are rendered very slowly. Tables should only be used to display tabular data.

**Example:**

<!DOCTYPE html>

<html>

<head>

    <style>

        \* {

            box-sizing: border-box;

        }

        .header {

            border: 1px solid red;

            padding: 15px;

        }

        .footer {

            border: 1px solid red;

            padding: 15px;

            clear: both;

        }

        .menu {

            width: 25%;

            float: left;

            padding: 15px;       /\* Without box-sizing, padding will make width not 25% anymore \*/

            border: 1px solid red;

        }

        .main {

            width: 75%;

            float: left;

            padding: 15px;      /\* Without box-sizing, padding will make width not 75% anymore \*/

            border: 1px solid red;

        }

    </style>

</head>

<body>

    <div class="header">

        <h1>Chania</h1>

    </div>

    <div class="menu">

        <ul>

            <li>The Flight</li>

            <li>The City</li>

            <li>The Island</li>

            <li>The Food</li>

        </ul>

    </div>

    <div class="main">

        <h1>The City</h1>

        <p>Chania is the capital of the Chania region on the island of Crete.</p>

        <p>Resize the browser window to see how the content respond to the resizing.</p>

    </div>

    <div class="footer">

        <center>

            Address ...

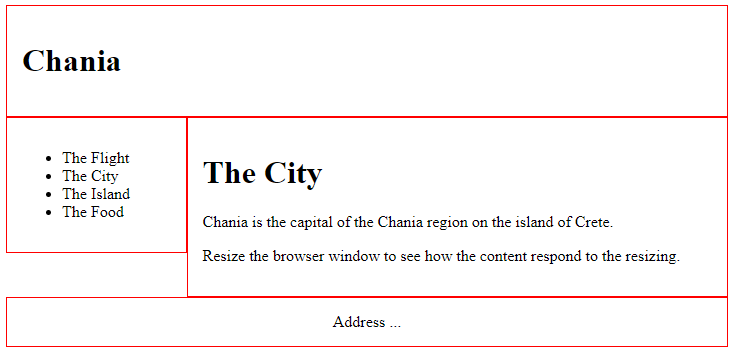
        </center>

    </div>

</body>

</html>

Output:



Responsive Grid View: [Responsive Web Design Grid (w3schools.com)](https://www.w3schools.com/css/css_rwd_grid.asp)

### CSS Flexbox

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

Learn more about flexbox in our [CSS Flexbox](https://www.w3schools.com/css/css3_flexbox.asp) chapter.

## CSS Properties

The CSS float property specifies how an element should float.

[CSS Layout - float and clear (w3schools.com)](https://www.w3schools.com/css/css_float.asp)

The CSS clear property specifies what elements can float beside the cleared element and on which side.

[CSS Layout - clear and clearfix (w3schools.com)](https://www.w3schools.com/css/css_float_clear.asp)

[CSS Layout - Float Examples (w3schools.com)](https://www.w3schools.com/css/css_float_examples.asp)

# Semantic Elements

Semantic elements = Elements with a meaning.

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

Examples:

* Non-semantic elements (Tells nothing about its content): <div> and <span>
* Semantic elements (Clearly defines its content): They are described in following sections:

All examples: [HTML Semantic Elements (w3schools.com)](https://www.w3schools.com/html/html5_semantic_elements.asp)

## HTML <section> Element

Defines a section in a document. A section is a thematic **grouping of content, typically with a heading**.

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

## HTML <article> Element

Specifies **independent, self-contained content**.

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

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

* Forum post
* Blog post
* Newspaper article

## HTML <header> Element

Represents a container for introductory content or a set of navigational links.

It typically contains:

* One or more heading elements (<h1> - <h6>)
* Logo or icon
* Authorship information

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

## HTML <footer> Element

Defines a footer for a document or section.

It typically contains:

* Authorship information
* Copyright information
* Contact information
* Sitemap
* Back to top links
* Related documents

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

## HTML <nav> Element

Defines a set of navigation links.

Note: NOT all links of a document should be inside a <nav> element. It is intended only for major block of navigation links.

## HTML <aside> Element

Defines some content aside from the content it is placed in (like a sidebar).

It should be indirectly related to the surrounding content.

## HTML <main> Element

Specify the main content of the document

The content inside the <main> element should be unique to the document. It should not contain any content that is repeated across documents such as sidebars, navigation links, copyright information, site logos, and search forms.

Note: There must not be more than one <main> element in a document. Also, it must NOT be a descendant of an <article>, <aside>, <footer>, <header>, or <nav> element.

## HTML <figure> and <figcaption> Elements

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

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

## Others

|  |  |
| --- | --- |
| [<mark>](https://www.w3schools.com/tags/tag_mark.asp) | Defines marked/highlighted text |
| [<details>](https://www.w3schools.com/tags/tag_details.asp) | Defines additional details that the user can view or hide |
| [<summary>](https://www.w3schools.com/tags/tag_summary.asp) | Defines a visible heading for a <details> element |
| [<time>](https://www.w3schools.com/tags/tag_time.asp) | Defines a date/time |

# JavaScript in HTML

## What Is Script in HTML?

A script is a small piece of program that can add interactivity to your website. You can write various small functions (called event handlers) using a script and then trigger those functions using HTML attributes. For example, a script could generate a pop-up alert box message when a button is clicked, or provide a dropdown menu.

Nowadays, only JavaScript and associated frameworks are being used by most of the web developers.

You can keep JavaScript code in a separate file and then include it wherever it's needed, or you can define functionality inside HTML document itself. Let's explore both ways.

## Different Ways to Use JavaScript in HTML

### External JavaScript

If you are going to define a functionality which will be used in various HTML documents, then it's always recommended to keep that functionality in a separate JavaScript file and then include that file in your HTML documents.

A JavaScript file will have extension as .js and it will be included in HTML files using <script> tag.

Consider we define a small function using JavaScript in script.js which has following code:

function ShowGreeting() {

    alert("Hello, World");

}

Now let's make use of the above external JavaScript file in the following HTML document:

<!DOCTYPE html>

<html>

   <head>

      <title>JavaScript External Script</title>

      <script src = "/html/script.js" type = "text/javascript"/></script>

   </head>

   <body>

      <input type = "button" onclick = "ShowGreeting();" name="ok" value="Click Me" />

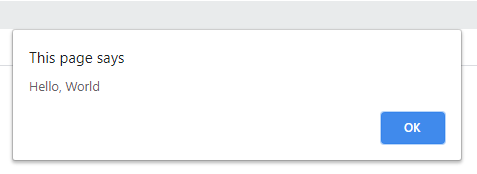
   </body>

</html>

Output:



When you click on this button, an alert message box is displayed:



### Internal Script

You can re-write the above example by inserting the JavaScript script code directly into your HTML document. There is no restriction where you can put your source code in the HTML document, but usually we keep it in header of the document using <script> tag.

<!DOCTYPE html>

<html>

   <head>

      <title>JavaScript Internal Script</title>

      <script type = "text/JavaScript">

         function ShowGreeting() {

            alert("Hello, World");

         }

      </script>

   </head>

   <body>

      <input type = "button" onclick = "ShowGreeting();" name="ok" value="Click Me" />

   </body>

</html>

## How to Choose Default Scripting Language

There may be a situation when you will include multiple script files and ultimately using multiple <script> tags. You can specify a default scripting language for all your script tags. This saves you from specifying the language every time you use a script tag within the page.

Below is the example:

<meta http-equiv = "Content-Script-Type" content = "text/JavaScript" />