**Full stack syllabus**

**HTML**:

**Introduction to frontend:**

The “front-end languages” live in the browser. After you type an address into the address bar at the top and hit the enter/return key, your browser will receive at least an HTML file from the web server. That file will likely tell the browser to request a CSS file and a JavaScript file as well (probably many more than one, but we’ll keep it simple).

Each of these languages performs a separate but very important function and they work harmoniously together to determine how the web page is STRUCTURED (HTML), how it LOOKS (CSS), and how it FUNCTIONS (JavaScript). And keep in mind that your *browser* handles figuring out how to make these files into a functioning web page (not the server).

**Structure of HTML**:

An HTML Document is mainly divided into two parts:

* **HEAD**: This contains the information about the HTML document. For Example, Title of the page, version of HTML, Meta Data etc.
* **BODY**: This contains everything you want to display on the Web Page.

Let us now have a look on the basic structure of HTML. That is the code which is must for every webpage to have:

<!DOCTYPE html>

<html>

<head>

<title>

</title>

</head>

<body>

</body>

</html>

Every Webpage must contain this code. Below is the complete explanation of each of the tags used in the above piece of HTML code:

**<!DOCTYPE html>:** This tag is used to tells the HTML version. This currently tells that the version is HTML 5.

**<html>:** This is called HTML root element and used to wrap all the code.

**<head>:** Head tag contains metadata, title, page CSS etc. All the HTML elements that can be used inside the <head> element are:

* <style>
* <title>
* <base>
* <noscript>
* <script>
* <meta>

We will learn about these in details later.

**<body>:** Body tag is used to enclosed all the data which a web page has from texts to links. All of the content that you see rendered in the browser is contained within this element.

**Tags and attribute in HTML**

* HTML Attributes:
* The href Attribute. The <a> tag defines a hyperlink. ...
* The src Attribute. The <**img**> tag is used to embed an image in an HTML page. ...
* The width and height Attributes. ...
* The alt Attribute. ...
* The style Attribute. ... 
* The lang Attribute. ...
* The **title** Attribute. ...
* All HTML elements can have **attributes**
* Attributes provide **additional information** about elements
* Attributes are always specified in **the start tag**
* Attributes usually come in name/value pairs like: **name="value"**

The href Attribute

The <a> tag defines a hyperlink. The href attribute specifies the URL of the page the link goes to:

Example: <a href="https://www.deviktech.com">Visit Deviktech</a>

## 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">

**The width and height Attributes**

The <img> tag should also contain the width and height attributes, which specifies 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 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">

<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:

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

You will learn more about styles in our [HTML Styles chapter](https://www.w3schools.com/html/html_styles.asp).

**The lang Attribute**

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

The following example specifies English as the language:

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

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

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

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

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

## The title Attribute

The title attribute defines some extra information about an element.

The value of the title attribute will be displayed as a tooltip when you mouse over the element

Example:

<p title="I'm a tooltip">This is a paragraph.</p>

**We Suggest: Always Use Lowercase Attributes**

The HTML standard does not require lowercase attribute names.

The title attribute (and all other attributes) can be written with uppercase or lowercase like **title** or **TITLE**.

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

**We Suggest: Always Quote Attribute Values**

The HTML standard does not require quotes around attribute values.

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

### **Good:**

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

### **Bad:**

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

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

Example:

<p title=About deviktech>

**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">

**Description of tags headings paragraph style and comments:**

**Tags:**

|  |  |
| --- | --- |
| <!**DOCTYPE** HTML> | The very first thing is the doctype declaration of the document. In this case HTML5. You should always use this. |
| <**html** *lang*="en"> | All this is doing is saying that this is the start of a HTML document. Additionally it specifies the language of the document, which is English. |
| <**head**> | Marks the start of the part of a HTML document called the head. This section is not visible on the webpage but it carries important information about the document. |
| <**meta** *charset*="UTF-8"> | Declaration of the character encoding: UTF 8 |
| <**title**>My First Web Page</**title**> | The page title can be seen in the browser window tab and when you bookmark the site in your favorites. You can put anything you want to in between the title tags. We also see something new here too. The last tag **</title>**, means that the tag title now ends. You can end any tag by putting a backslash (**/**) after the less than sign (**<**). |
| </**head**> | Closing of the head section |
| <**body**> | We begin the body section. Here is all the stuff is that will be visible on the page. |
| Hello World! | The part visible in the web browser |
| </**body**> | Closing body tag. |
| </**html**> | Ending the document |

Headings:

**HTML** defines six levels of **headings**. A **heading** element implies all the font changes, paragraph breaks before and after, and any white space necessary to render the **heading**. The **heading** elements are H1, H2, H3, H4, H5, and H6 with H1 being the highest (or most important) level and H6 the least.

Paragraph:

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 Deviktech training center</p>

Style:

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

Example:

Deviktech

Deviktech

Deviktech

Comment:

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

<!-- Write your comments here -->

Block and inline:

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

There are two display values: block and inline.

**Block-level Elements**

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

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

Example:

<div>Hello World</div>

Here are the block-level elements in HTML:

[<address>](https://www.w3schools.com/tags/tag_address.asp)

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

**Inline Elements**

An inline element does not start on a new line and it only takes up as much width as necessary.

This is a <span> element inside a paragraph.

Example

<span>Hello World</span>

Here are the inline elements in HTML:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

## The <div> Element

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

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

When used together with CSS, the <div> element can be used to style blocks of content:

Example:

<div style="background-color:black;color:white;padding:20px;">  
  <h2>London</h2>  
  <p>London is the capital city of England. It is the most populous city in the United Kingdom, with a metropolitan area of over 13 million inhabitants.</p>  
</div>

The <span> Element

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

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

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

Example:

<p>My mother has <span style="color:blue;font-weight:bold">blue</span> eyes and my father has <span style="color:darkolivegreen;font-weight:bold">dark green</span> eyes.</p>

## Chapter Summary

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

**HTML Tags**

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

How to use images ,Iframe ,links:

The **iframe** needs to be given a name. This example uses three [**<a> tags**](http://www.littlewebhut.com/html/a_tag/) which are used to select which image to display. The **iframe** name is needed by the [**<a> tags**](http://www.littlewebhut.com/html/a_tag/) as a destination for the images. The [**<a> tag**](http://www.littlewebhut.com/html/a_tag/) uses its **target attribute** to point to the **iframe**. The [**<a> tag**](http://www.littlewebhut.com/html/a_tag/) **target attribute** and the **iframe** name need to match. The **iframe** name is set using the **iframe name attribute** and the **iframe id attribute**.  
The syntax for the **<iframe> tag attributes** is:

**name="imgbox" id="imgbox"**

The syntax for the **<a> tag attribute** is:

**target="imgbox"**

Classes and id:

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

Multiple HTML elements can share the same class.

**Using The class Attribute**

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

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

Example:

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

</body>

</html>

Id:

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

You cannot have more than one element with the same id in an HTML document.

**Using The id Attribute**

The id attribute specifies a unique id for an HTML element. The value of the id attribute must be unique within the HTML document.

The id attribute is used to point to a specific style declaration in a style sheet. It is also used by JavaScript to access and manipulate the element with the specific id.

The syntax for id is: write a hash character (#) character, followed by an id name. Then, define the CSS properties within curly braces {}.

In the following example we have an <h1> element that points to the id name "myHeader". This <h1> element will be styled according to the #myHeader style definition in the head section:

Example:

<!DOCTYPE html>  
<html>  
<head>  
<style>  
#myHeader {  
  background-color: lightblue;  
  color: black;  
  padding: 40px;  
  text-align: center;  
}  
</style>  
</head>  
<body>  
  
<h1 id="myHeader">My Header</h1>  
  
</body>  
</html>

Forms:

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

### **Example**

Top of Form

First name:  
  
Last name:  
  
  


Bottom of Form

**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

How to use media ,video ,audio, plugins, youtube:

Multimedia on the web is sound, music, videos, movies, and animations.

What is Multimedia?

Multimedia comes in many different formats. It can be almost anything you can hear or see, like images, music, sound, videos, records, films, animations, and more.

Web pages often contain multimedia elements of different types and formats.

Browser Support

The first web browsers had support for text only, limited to a single font in a single color.

Later came browsers with support for colors, fonts, images, and multimedia!

Multimedia Formats

Multimedia elements (like audio or video) are stored in media files.

The most common way to discover the type of a file, is to look at the file extension.

Multimedia files have formats and different extensions like: .wav, .mp3, .mp4, .mpg, .wmv, and .avi.

Common Video Formats

|  |  |
| --- | --- |
| Videoformats | There are many video formats out there.  The MP4, WebM, and Ogg formats are supported by HTML.  The MP4 format is recommended by YouTube. |

VEDIO:

The HTML <video> element is used to show a video on a web page.

**Example**

<video width="320" height="240" controls>  
  <source src="movie.mp4" type="video/mp4">  
  <source src="movie.ogg" type="video/ogg">  
</video>

AUDIO:

## 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>

PLUG-IN:

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

YOUTUBE:

The easiest way to play videos in HTML, is to use YouTube.

**Struggling with Video Formats?**

Earlier in this tutorial, you have seen that you might have to convert your videos to different formats to make them play in all browsers.

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>



**CSS:**

Introduction,syntax,selectors:

**CSS Syntax**

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



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

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

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

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

### **Example**

In this example all <p> elements will be center-aligned, with a red text color:

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

#### **Example Explained**

* p is a **selector** in CSS (it points to the HTML element you want to style: <p>).
* color is a property, and red is the property value
* text-align is a property, and center is the property value

Colors,background:

Colors are specified using predefined color names, or RGB, HEX, HSL, RGBA, HSLA values.

**CSS Color Names**

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

Tomato

Orange

DodgerBlue

MediumSeaGreen

Gray

SlateBlue

Violet

.**CSS Background Color**

You can set the background color for HTML elements:

Hello World

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

### **Example**

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

# **CSS Backgrounds**

The CSS background properties are used to define the background effects for elements.

In these chapters, you will learn about the following CSS background properties:

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

## CSS background-color

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

### **Example**

The background color of a page is set like this:

body {  
  background-color: lightblue;  
}

Borders ,Margins,Padding,Height:

Border:

# **CSS Borders**

CSS Border Properties

The CSS border properties allow you to specify the style, width, and color of an element's border.

I have borders on all sides.

I have a red bottom border.

I have rounded borders.

I have a blue left border.

CSS Border Style

The border-style property specifies what kind of border to display.

The following values are allowed:

* dotted - Defines a dotted border
* dashed - Defines a dashed border
* solid - Defines a solid border
* double - Defines a double border
* groove - Defines a 3D grooved border. The effect depends on the border-color value
* ridge - Defines a 3D ridged border. The effect depends on the border-color value
* inset - Defines a 3D inset border. The effect depends on the border-color value
* outset - Defines a 3D outset border. The effect depends on the border-color value
* none - Defines no border
* hidden - Defines a hidden border

The border-style property can have from one to four values (for the top border, right border, bottom border, and the left border).

### **Example**

Demonstration of the different border styles:

p.dotted {border-style: dotted;}  
p.dashed {border-style: dashed;}  
p.solid {border-style: solid;}  
p.double {border-style: double;}  
p.groove {border-style: groove;}  
p.ridge {border-style: ridge;}  
p.inset {border-style: inset;}  
p.outset {border-style: outset;}  
p.none {border-style: none;}  
p.hidden {border-style: hidden;}  
p.mix {border-style: dotted dashed solid double;}

Result:

A dotted border.

A dashed border.

A solid border.

A double border.

A groove border. The effect depends on the border-color value.

A ridge border. The effect depends on the border-color value.

An inset border. The effect depends on the border-color value.

An outset border. The effect depends on the border-color value.

No border.

A hidden border.

A mixed border.

Margins: 

# **CSS Margins**

This element has a margin of 70px.

CSS Margins

The CSS margin properties are used to create space around elements, outside of any defined borders.

With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

Margin - Individual Sides

CSS has properties for specifying the margin for each side of an element:

* margin-top
* margin-right
* margin-bottom
* margin-left

All the margin properties can have the following values:

* auto - the browser calculates the margin
* length - specifies a margin in px, pt, cm, etc.
* % - specifies a margin in % of the width of the containing element
* inherit - specifies that the margin should be inherited from the parent element

**Tip:** Negative values are allowed.

### **Example**

Set different margins for all four sides of a <p> element:

p {  
  margin-top: 100px;  
  margin-bottom: 100px;  
  margin-right: 150px;  
  margin-left: 80px;  
}

Padding:

## CSS Padding

The CSS padding properties are used to generate space around an element's content, inside of any defined borders.

With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

Padding - Individual Sides

CSS has properties for specifying the padding for each side of an element:

* padding-top
* padding-right
* padding-bottom
* padding-left

All the padding properties can have the following values:

* length - specifies a padding in px, pt, cm, etc.
* % - specifies a padding in % of the width of the containing element
* inherit - specifies that the padding should be inherited from the parent element

**Note:** Negative values are not allowed.

### **Example**

Set different padding for all four sides of a <div> element:

div {  
  padding-top: 50px;  
  padding-right: 30px;  
  padding-bottom: 50px;  
  padding-left: 80px;  
}

Height ,width:

CSS Setting height and width

The height and width properties are used to set the height and width of an element.

The height and width properties do not include padding, borders, or margins. It sets the height/width of the area inside the padding, border, and margin of the element.

CSS height and width Values

The height and width properties may have the following values:

* auto - This is default. The browser calculates the height and width
* length - Defines the height/width in px, cm etc.
* % - Defines the height/width in percent of the containing block
* initial - Sets the height/width to its default value
* inherit - The height/width will be inherited from its parent value

CSS height and width Examples

This element has a height of 200 pixels and a width of 50%

### **Example**

Set the height and width of a <div> element:

div {  
  height: 200px;  
  width: 50%;  
  background-color: powderblue;  
}

Text,fonts,links,max-width,overflow,float,align:

Max-width:

## Setting max-width

The max-width property is used to set the maximum width of an element.

The max-width can be specified in length values, like px, cm, etc., or in percent (%) of the containing block, or set to none (this is default. Means that there is no maximum width).

The problem with the <div> above occurs when the browser window is smaller than the width of the element (500px). The browser then adds a horizontal scrollbar to the page.

Using max-width instead, in this situation, will improve the browser's handling of small windows.

**Tip:** Drag the browser window to smaller than 500px wide, to see the difference between the two divs!

This element has a height of 100 pixels and a max-width of 500 pixels.

**Note:** The value of the max-width property overrides width.

### **Example**

This <div> element has a height of 100 pixels and a max-width of 500 pixels:

div {  
  max-width: 500px;  
  height: 100px;  
  background-color: powderblue;  
}

Text:

## Text Color

The color property is used to set the color of the text. The color is specified by:

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

Look at [CSS Color Values](https://www.w3schools.com/cssref/css_colors_legal.asp) for a complete list of possible color values.

The default text color for a page is defined in the body selector.

### **Example**

body {  
  color: blue;  
}  
  
h1 {  
  color: green;  
}

**Text Color and Background Color**

In this example, we define both the background-color property and the color property:

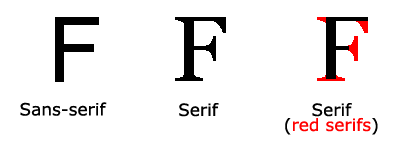
### **Example**

body {  
  background-color: lightgrey;  
  color: blue;  
}  
  
h1 {  
  background-color: black;  
  color: white;

Font:

The CSS font properties define the font family, boldness, size, and the style of a text.

Difference Between Serif and Sans-serif Fonts



**CSS Font Families**

In CSS, there are two types of font family names:

* **generic family** - a group of font families with a similar look (like "Serif" or "Monospace")
* **font family** - a specific font family (like "Times New Roman" or "Arial")

**Font Family**

The font family of a text is set with the font-family property.

The font-family property should hold several font names as a "fallback" system. If the browser does not support the first font, it tries the next font, and so on.

Start with the font you want, and end with a generic family, to let the browser pick a similar font in the generic family, if no other fonts are available.

**Note**: If the name of a font family is more than one word, it must be in quotation marks, like: "Times New Roman".

More than one font family is specified in a comma-separated list:

### **Example**

Specify the font for three paragraphs:

.serif {  
  font-family: "Times New Roman", Times, serif;  
}  
  
.sansserif {  
  font-family: Arial, Helvetica, sans-serif;  
}  
  
.monospace {  
  font-family: "Lucida Console", Courier, monospace;  
}

### **Example**

Specify the "Impact" font for a paragraph:

p.impact {  
  font-family: Impact, Charcoal, sans-serif;  
}

Links:

With CSS, links can be styled in different ways.

[Text Link](javascript:void(0)) [Text Link](javascript:void(0)) [Link Button](javascript:void(0)) [Link Button](javascript:void(0))

## Styling Links

Links can be styled with any CSS property (e.g. color, font-family, background, etc.).

**Example**

a {  
  color: hotpink;  
}

Overflow:

The CSS overflow property controls what happens to content that is too big to fit into an area.

CSS Overflow

The overflow property specifies whether to clip the content or to add scrollbars when the content of an element is too big to fit in the specified area.

The overflow property has the following values:

* visible - Default. The overflow is not clipped. The content renders outside the element's box
* hidden - The overflow is clipped, and the rest of the content will be invisible
* scroll - The overflow is clipped, and a scrollbar is added to see the rest of the content
* auto - Similar to scroll, but it adds scrollbars only when necessary

Float:

The CSS float property specifies how an element should float.

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

The float Property

The float property is used for positioning and formatting content e.g. let an image float left to the text in a container.

The float property can have one of the following values:

* left - The element floats to the left of its container
* right - The element floats to the right of its container
* none - The element does not float (will be displayed just where it occurs in the text). This is default
* inherit - The element inherits the float value of its parent

In its simplest use, the float property can be used to wrap text around images.

Example - float: right;

Align:

## Center Align Elements

To horizontally center a block element (like <div>), use margin: auto;

Setting the width of the element will prevent it from stretching out to the edges of its container.

The element will then take up the specified width, and the remaining space will be split equally between the two margins:

This div element is centered.

### **Example**

.center {  
  margin: auto;  
  width: 50%;  
  border: 3px solid green;  
  padding: 10px;  
}

**Note:** Center aligning has no effect if the width property is not set (or set to 100%).

**Center Align Text**

To just center the text inside an element, use text-align: center;

This text is centered.

### **Example**

.center {  
  text-align: center;  
  border: 3px solid green;  
}

Flex:

CSS Flexbox Layout Module

Before the Flexbox Layout module, there were four layout modes:

* Block, for sections in a webpage
* Inline, for text
* Table, for two-dimensional table data
* Positioned, for explicit position of an element

The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure without using float or positioning.

Media query:

CSS2 Introduced Media Types

The @media rule, introduced in CSS2, made it possible to define different style rules for different media types.

Examples: You could have one set of style rules for computer screens, one for printers, one for handheld devices, one for television-type devices, and so on.

Unfortunately these media types never got a lot of support by devices, other than the print media type.

CSS3 Introduced Media Queries

Media queries in CSS3 extended the CSS2 media types idea: Instead of looking for a type of device, they look at the capability of the device.

Media queries can be used to check many things, such as:

* width and height of the viewport
* width and height of the device
* orientation (is the tablet/phone in landscape or portrait mode?)
* resolution

Using media queries are a popular technique for delivering a tailored style sheet to desktops, laptops, tablets, and mobile phones (such as iPhone and Android phones).

Browser Support

The numbers in the table specifies the first browser version that fully supports the @media rule.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Property |  |  |  |  |  |
| @media | 21.0 | 9.0 | 3.5 | 4.0 | 9.0 |

Media Query Syntax

A media query consists of a media type and can contain one or more expressions, which resolve to either true or false.

@media not|only *mediatype*and(*expressions*) { *CSS-Code;*}