

Chapter 2

CSS – part 1

Definition

CSS (Cascading Style Sheets) defines the appearance of the HTML elements. A style sheet (CSS) is a text file that contains rules that determine how elements should be displayed by the browser.

The current version is **CSS3**.

CSS rules

Each CSS rule has two parts:

- **The selector:** determines which elements are affected, it can also be a class or an id

	HTML	CSS
An element type	<code><p> a red paragraph </p></code>	<code>p { color: red; }</code>
A class	<code><p class="center"> Red and center-aligned paragraph </p></code>	<code>.center { text-align: center; color: red; }</code>
An id	<code><p id="para1">Hello World!</p></code>	<code>#para1 { text-align: center; color: red; }</code>
*	<code><p> Every element will be affected </p></code>	<code>* { text-align: center; color: blue; }</code>

- **The declaration block:**

It is a list of property-value pairs, where

a property is separated from its value by a colon :

And the property-value pairs are separated by semi-colons ;

For readability, each declaration should be placed on a separate line

Important: ids can be used only once inside a page, while classes can be used multiple times. Then, you should use the id when you want to target a specific HTML element with a CSS rule. You should use classes when you want a CSS rule to target different HTML elements.

The CSS Grouping element Selector

<pre>h1 { text-align: center; color: red; } h2 { text-align: center; color: red; } p { text-align: center; color: red; }</pre>	<pre>h1, h2, p { text-align: center; color: red; }</pre>
--	--

The grouping selector selects all the HTML elements with the same style definitions and group the selectors, to minimize the code. See [HERE](#) and [HERE](#)

CSS definition

- **Inline styles:** Within a single HTML tag
- **Internal/Global/ Embedded styles:** In the <head> section
- **External files:** spirit .CSS file

1- Inline styles

An inline style may be used to apply a unique style for a single element. To use inline styles, add the **style** attribute to the relevant element. The style attribute can contain any CSS property.

Inline styles

```
<h1 style="color:blue; border:ridge"> Inline styles </h1>
<p style="margin-left:10%; background:#ffffcc">some text
. . . some text</p>
```

some text some text some
text some text some text
some text some text

But this is **discouraged** as it breaks the separation of presentation and content design principle.

2- Internal/Global/Embedded CSS

An internal style sheet may be used if one single HTML page has a unique style. The internal style is defined inside the `<style>` element, which is inside the **head** section.

`<head>`

```
<title>Page 1</title>
<style>
  p {
    color: yellow;
    background-color: blue;
    text-align:center;
  }
</style>
</head>
<body>
  <p>paragraph 1</p>
</body>
```

paragraph 1

Notes:

- This is also discouraged for the same reason.

3- External CSS

you can change the look of an entire website by changing just one .css file!

Each HTML page must include a reference to that external style sheet file inside the `<link>` element, which is inside the **head** section.

.HTML	.CSS
<pre><head> <meta charset="utf-8"> <title>Page 1</title> <link " rel="stylesheet" href="index.css" type="text/css"> </head></pre>	<pre>body { background-color: lightblue; } h1 { color: navy; margin-left: 20px; }</pre>

Notes:

- **rel**, the type of the link
- **href**, the hyperlink to the file
- **type**, type of data stored
- no space between the property value and the unit

Multiple stylesheets

If some properties have been defined for the same selector (element) in different style sheets, the value from the last read style sheet will be used.

Example:

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
<style>
h1 {
  color: orange;
}
</style>
</head>
```

Which one will be applied?

Cascading order:

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

All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

- 1) Inline style (inside an HTML element)
- 2) External and internal style sheets (in the head section)
- 3) Browser default

So, an inline style has the **highest** priority, and will override external and internal styles and browser defaults. See [Here](#)

Also,

The law of specificity states that the rule with the most specific selector will be applied.

html	css	rendering
<pre><h1>Header A</h1> <h1 class="classH">Header B </h1> <h1 id="headerC"> Header C </h1></pre> <p>Notes:</p> <ul style="list-style-type: none">- The first rule with the h1 selector applies to header A.- For Header B, the first rule is masked by the second rule with the selector h1.classH because selectors with defined classes are more specific.- For Header A, the last selector with the id specified is applied because selectors with id defined are more specific than selectors with class defined.	<pre>h1 { color: red; } .classH { color: green; } #headerC { color: black; } #headerC { color: blue; }</pre>	<p>Header A</p> <p>Header B</p> <p>Header C</p>

CSS Comments

You can add comments to CSS files using (`/* text */`) and can span multiple lines:
`/* The following rules define the style of the top-level
elements */`

```
header {  
    color: red; /* Makes text color red */  
}
```

Styling text

a. The font-family property

There are types of font families for each generic:

Generic family	Family name	Description
Serif	Times New Roman Georgia	Serif fonts have small lines at the ends on some characters
Sans-serif	Arial Verdana	"Sans" means without - these fonts do not have the lines at the ends of characters
Monospace	Courier New Lucida Console	All monospace characters have the same width
Cursive	<i>Brush Script MT</i> <i>Lucida Handwriting</i>	imitate human handwriting
Fantasy	Copperplate Papyrus	decorative/playful fonts

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

font-family: <family name> [<generic family>]

Example,

font-family: "Times New Roman", Times, serif;

The browser first looks for the "*Times New Roman*" font–If this is not on the system, it looks for the *Times* font, Last resort: the browser uses the *generic serif* font

b. Other properties are available:

font-style: normal/italic/oblique

font-weight: normal/bold/bolder/lighter

font-size: small/medium/large/smaller/larger

text-decoration: none/underline/overline/line-through

text-transformation: none/capitalize/uppercase/lowercase

text-align: left/right/center/justify

text-indentation: length/percentage

Using the `vw` unit allows to have a responsive design.
`vw`(Viewport) is the browser window size. `1vw = 1%` of viewport width.

`font-size:10vw;`

See [HERE](#), [HERE](#) and [HERE](#)

CSS colors

set the background color for HTML elements: `background-color:DodgerBlue;`

set the color of text: `color:#120421;`

set the color of borders: `border:2px solid Tomato;`

colors can also be specified using `RGB` values, `HEX` values, `HSL` values, `RGBA` values, and `HSLA` values:

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

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

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

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

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

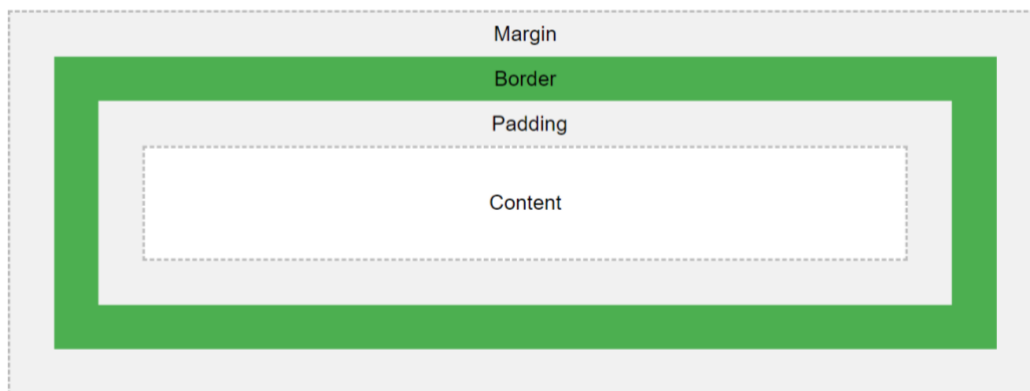
See [HERE](#)

Learn something interesting [HERE](#)

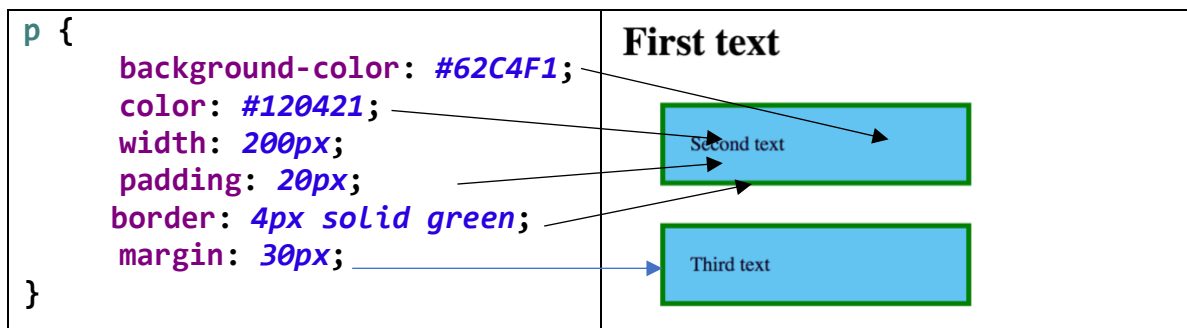
The CSS Box Model

Every HTML element can be viewed as a box. It consists of:

a `margin`, `borders`, a `padding`, and the actual `content`. The image below illustrates the box model:



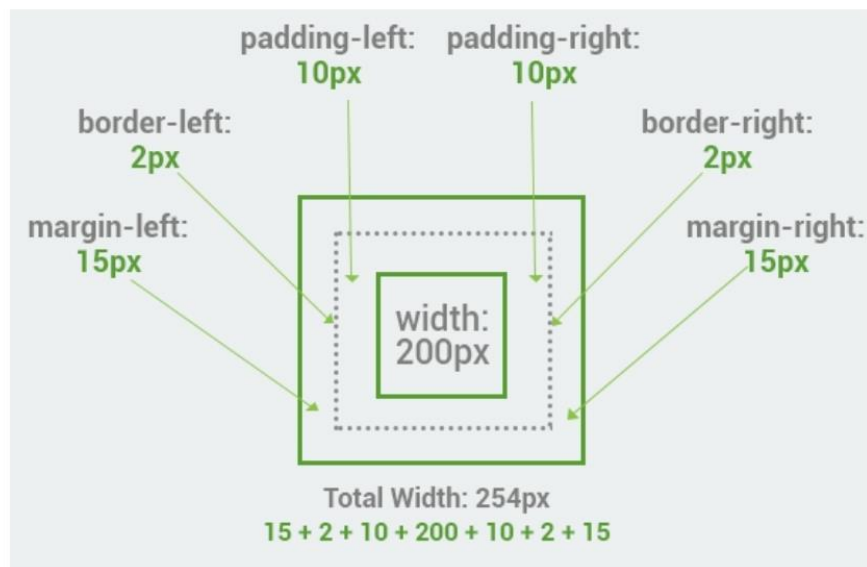
- **Content:** where the page content appear
- **Padding:** an area around the content. The padding is transparent
- **Border:** A border that goes around the padding and content
- **Margin:** an area outside the border. The margin is transparent



Notes:

- **color**: will color the text
- **background-color**: will covers the content area, as well as the padding.
- **border**: will put frame the content area, as well as the padding. See [HERE](#)
- **width** and height properties of an element are applied to the content area. You can set the width or height using %:

More Description



CSS Positioning property:

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the **position** property is set first. There are five different position values:

-static -relative -fixed -absolute -sticky

```

p{
  position: value;
}

```

See [HERE](#) and [HERE](#)

- Static Positioning

HTML elements are positioned **static** by default. A static positioned element is always positioned according to the **normal flow of the page**.

The properties **top**, **right**, **bottom**, and **left** do not have any effects.

- Relative Positioning

A relative positioned element is positioned **relative to its normal position**.

The properties **top**, **right**, **bottom**, and **left** can be used to specify how the rendered box will be shifted.

- Fixed Positioning

An element with a fixed position is positioned **relative to the browser window** and will not move even if the window is scrolled.

The position can be specified using one or more of the properties **top**, **right**, **bottom**, and **left**.

Fixed positioned elements are removed from the normal flow. Fixed positioned elements can overlap other elements.

- Absolute Positioning

An element with position: absolute; is positioned relative to the **nearest positioned ancestor** (instead of positioned relative to the viewport, like fixed).

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.


CSS floating and clearing 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

<pre><p> This paragraph has an image that is floated to the right. </p></pre>	<pre>img { float: right; }</pre>	<div>This paragraph has an image that is floated to the right.</div> 
---	--------------------------------------	---

It is **highly recommended to add a margin** to images so that the text does not get too close to the image. If you want your text to be easily read, you should always add a few pixels between words and borders, images, and other content.

- **Clearing the Float**

The clear property specifies what elements can float beside the cleared element and on which side. The clear property can have one of the following values:

left - No floating elements allowed on the left side

right - No floating elements allowed on the right side

both - No floating elements allowed on either the left or the right side

none - Allows floating elements on both sides. This is default

See [HERE](#)

The overflow Property

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

See [HERE](#)