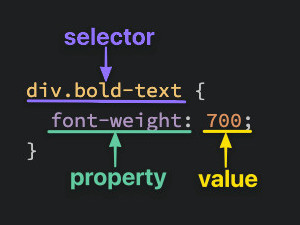
**CSS 01**

Basic Syntax – made up of a selector; a semi-colon separated list of declarations, with each declarations made up of a property:value pait.



<div> is a basic HTML element. It is an empty container. Generally, its best to use other tags (e.g. <h1>, <p>) for content, but in some cases where you only need a container for other elements, you use <div> for simplicity.

**Selector**

refer to the HTML elements to which CSS rule apply.

**Univeral Selector [ \* ]**

select elements of any type, eg:

*\* {*

*color: purple;*

*}*

**Type Selector [ e.g.:div ]**

select all elements of the given type, syntax is a name of the element

*<!-- index.html -->*

*<div>Hello, World!</div>*

*<div>Hello again!</div>*

*<p>Hi...</p>*

*<div>Okay, bye.</div>*

*/\* styles.css \*/*

*div {*

*color: white;*

*}*

all the <div> elements would be selected, while <p> wouldn’t be

**Class Selector [ .xxx ]**

select all elements with the given class

*<!-- index.html -->*

*<div class="alert-text">*

*Please agree to our terms of service.*

*</div>*

*/\* styles.css \*/*

*.alert-text {*

*color: red;*

*}*

syntax: a period [**.**]immediately followed by case-sensitive value of the class attribute. Muliple class can be added to a single element as a **space-separated** list. Eg: *class=”alert-text severe-alert”.* As whitespace is used to sepearte class name, mulit-worded names should use a hyphen instead.

**ID selector [ #xxx ]**

similar to class selector, select an element with the given ID

*<!-- index.html -->*

*<div id="title">My Awesome 90's Page</div>*

*/\* styles.css \*/*

*#title {*

*background-color: red;*

*}*

instead of [**.**], [**#**] is used instead, immediately followed by the case-sensitive value of the ID attribute.

Major difference is an element can only have **one** ID. An ID cannot be repeated on a single page, the ID attribute should not contain any whitespace.

**Grouping Selector**

group two selector with shared declarations

From:

*.read {*

*color: white;*

*background-color: black;*

*/\* several unique declarations \*/*

*}*

*.unread {*

*color: white;*

*background-color: black;*

*/\* several unique declarations \*/*

*}*

To:

*.read,*

*.unread {*

*color: white;*

*background-color: black;*

*}*

*.read {*

*/\* several unique declarations \*/*

*}*

*.unread {*

*/\* several unique declarations \*/*

*}*

reduce repetition of declarations, makes it easier to edit either color or background-color for both classes at once.

**Chaining Selector**

chain them as a list without any separation

*<div>*

*<div class="subsection header">Latest Posts</div>*

*<p class="subsection preview">This is where a preview for a post might go.</p>*

*</div>*

*.subsection.header {*

*color: red;*

*}*

.subsection.header selects any elements that has both “subsection” & “header” classes.

*<div>*

*<div class="subsection header">Latest Posts</div>*

*<p class="subsection" id="preview">This is where a preview for a post might go.</p>*

*</div>*

*.subsection.header {*

*color: red;*

*}*

*.subsection#preview {*

*color: blue;*

*}*

can also be used to chain a class and an ID

In general, only one type selector can be chained.

**Descendant Combinator**

combine multiple selectors, other than grouping or chaining. There are four types of combinators

descendant combinator, represented in CSS by a single space between selectors. Only cause elements that match the last selector that also have an ancestor that matches the previous selector

.ancestor .child would select an element with class “child” only if it has an ancestor with the class “ancestor”.

*<!-- index.html -->*

*<div class="ancestor"> <!-- A -->*

*<div class="contents"> <!-- B -->*

*<div class="contents"> <!-- C -->*

*</div>*

*</div>*

*</div>*

*<div class="contents"></div> <!-- D -->*

*/\* styles.css \*/*

*.ancestor .contents {*

*/\* some declarations \*/*

*}*

in the above, only the first two elements with the “contents” class (B & C) would be selected, last element (D) won’t be

there is no limit to how many combinators you can add to a rule

**Properties**

Color & Background-color

both color properties can accept:

-color name: red, transparent

-HEX value: #1100ff

-RGB value: rgb(100, 0, 127)

-HSL value: hsl(15, 82%, 56%)

*p {*

*/\* hex example: \*/*

*color: #1100ff;*

*/\* rgb example: \*/*

*color: rgb(100, 0, 127);*

*/\* hsl example: \*/*

*color: hsl(15, 82%, 56%);*

*}*

<https://www.w3schools.com/>cssref/css\_colors\_legal.php

**Typography Basic & Text-Align**

**font-family**, can be a single value or a comma-separated list of values.

“font family name” like “Times New Roman” (uses quotes due to whitespaces between words)

“generic family name” like sans-serif (generic never use quotes)

if a browser cannot find or does not support the first font on the list, it will use the next one, until it finds a supported and valid font.

Its best practicce to include a list of values for this property, with a generic font family as a fallback.

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

**font-size**, set size of the font

*font-size: 22px*

**font-weight**, affects the boldness of text, value can be a keyword or a number between 1 & 1000

*font-weight: bold*

*font-weight: 700 (equivalent of bold)*

**text-align**, align text horizontally within an element, common keywords can be used

*text-align: center*

**Image Height and Width**

<img> element’s height and width values can be adjusted, its best to include both properties to <img> elements to reserve space on webpage

*img {*

*height: auto;*

*width: 500px;*

*}*

**The Cascade of CSS**

the cascade is what determines which rule actually get apllied to the HTML.

Specificity

a CSS declaration more specific will take precedence.

1. ID selector

2. Classs selector

3. Type selector

*<div class="main">*

*<div class="list subsection"></div>*

*</div>*

*/\* rule 1 \*/*

*.subsection {*

*color: blue;*

*}*

*/\* rule 2 \*/*

*.main .list {*

*color: red;*

*}*

color: red takes precedence as it has more class selector

*<div class="main">*

*<div class="list" id="subsection"></div>*

*</div>*

*/\* rule 1 \*/*

*#subsection {*

*color: blue;*

*}*

*/\* rule 2 \*/*

*.main .list {*

*color: red;*

*}*

color: blue takes precedence as ID beats class

*/\* rule 1 \*/*

*#subsection .list {*

*background-color: yellow;*

*color: blue;*

*}*

*/\* rule 2 \*/*

*#subsection .main .list {*

*color: red;*

*}*

color: red takes precedence as it has more class selector,

background-color: yellow will still be applied as there is no conflict

*/\* rule 1 \*/*

*.class.second-class {*

*font-size: 12px;*

*}*

*/\* rule 2 \*/*

*.class .second-class {*

*font-size: 24px;*

*}*

both have the same specificity (chaining selector vs descendant combinator)

*/\* rule 1 \*/*

*.class.second-class {*

*font-size: 12px;*

*}*

*/\* rule 2 \*/*

*.class > .second-class {*

*font-size: 24px;*

*}*

both have the same specificity (chaining selector vs child combinator)

*/\* rule 1 \*/*

*\* {*

*color: black;*

*}*

*/\* rule 2 \*/*

*h1 {*

*color: orange;*

*}*

rule 2 takes precedence ( type selector vs universal selector)

**Inheritance**

certain CSS properties when applied to the element, will be inherited by the element’s descendant, even if it is not explicitly written.

typography based properties (color, font-size, font-family, etc) are usually inherited, most other properties aren’t.

*<!-- index.html -->*

*<div id="parent">*

*<div class="child"></div>*

*</div>*

*/\* styles.css \*/*

*#parent {*

*color: red;*

*}*

*.child {*

*color: blue;*

*}*

child element will be blue as directly declared, regardess of the higher specificity of parent element

**Rule Order**

if multiple rules are conflicting and ties in specificity, the **last defined rule wins**

*/\* styles.css \*/*

*.alert {*

*color: red;*

*}*

*.warning {*

*color: yellow;*

*}*

an element with alert & warning classes, .warning rule will be applied

**Adding CSS to HTML**

External CSS

most common method, creating a separate file for the CSS and linking it inside of a HTML’s opening and closing <head> tags, with a self-closing <link> element

*<!-- index.html -->*

*<head>*

***<link rel="stylesheet" href="styles.css">***

*</head>*

*/\* styles.css \*/*

*div {*

*color: white;*

*background-color: black;*

*}*

*p {*

*color: red;*

*}*

first, the self-closing *<link>* element is added inside the *<head> </head>* tags.

the ***herf*** attr. is the location of the CSS file, an absolute URL or a URL relative to the location of the HTML file.

the ***rel***attr. sepcifies the relationship between the HTML file and the linked file.

inside the newly created *style.css* file, we have the selector (the *div* and *p*), followed by a part of opening the closing curly braces *{ }*, creating a “declaration block”.

declarations are placed inside the “declaration block”.

color: white – color is the property, white is the value.

**Internal CSS**

Internal CSS / embedded CSS, means the CSS is added within the HTML file itself.

all rules were placed inside a pair of opening and closing <style> </style> tags, placed inside <head> </head>. the <link> element is no longer need in this case.

*<head>*

*<style>*

*div {*

*color: white;*

*background-color: black;*

*}*

*p {*

*color: red;*

*}*

*</style>*

*</head>*

*<body>...</body>*

**Inline CSS**

add style direcrtly to HTML elements, tho not recommended

*<body>*

*<div style="color: white; background-color: black;">...</div>*

*</body>*

selector is not used, the styles are being added directly to the opening <div> tag

***style=*** newattr. with its value within the pair of quotation marks being the declaration

only use for adding a unique style to a single element, or it gets messy quickly