

# Programma “Frontend Developer”

## Modulo 1 - CSS

Docente: Diana Bernabei

# CSS

CSS è l'acronimo di Cascading Style Sheets, ed è un linguaggio utilizzato per definire la formattazione di documenti HTML.

Cascading Style Sheet sono appunto dei fogli di stile a cascata, perché quando si va a definire più volte lo stile di un tag verrà di volta in volta riassegnato un nuovo stile a quel tag.



# Dove scrivere codice CSS

- **inline** utilizzando l'attributo "style"
- nel tag style **interno** al documento
- in un file **esterno** con estensione .css e importando il file del documento con

```
<link rel="stylesheet" href="style.css">
```

# Applicare il CSS inline

```

<!DOCTYPE html>
<html>
<body>

<h1 style="color:blue;text-align:center;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>

</body>
</html>

```

Esistono tre modi per applicare CSS a HTML: Inline, interno ed esterno.

Gli stili incorporati vengono inseriti direttamente nei tag HTML utilizzando l'attributo style.

# Applicare il CSS internamente

Gli stili incorporati o interni vengono utilizzati per l'intera pagina. All'interno dell'elemento head, i tag di stile circondano tutti gli stili per la pagina.

```
<!DOCTYPE html>
<html>
<head>
<style>
  body {
    background-color: lightblue;
  }

  h1 {
    color: red;
    margin-left: 40px;
  }
</style>
</head>
<body>

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

</body>
</html>
```

# Applicare il CSS esternamente

index.html

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>

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

</body>
</html>
```

style.css

```
body {
  background-color: lightblue;
}

h1 {
  color: navy;
  margin-left: 20px;
}
```

## Come si commenta

Si utilizza `/* ... */` per inserire commenti che descrivono delle sezioni di CSS o che comunque aiuti a rendere il codice più comprensibile.

# Sintassi

## selettore { proprietà : valore ; }

Un set di regole CSS è composto da un selettore e un blocco di dichiarazione:

- Il selettore punta all'elemento HTML che si desidera modellare.
- Il blocco di dichiarazioni contiene una o più dichiarazioni separate da punti e virgola.

Ogni dichiarazione include un nome di proprietà CSS e un valore, separati da due punti.

Una dichiarazione CSS termina sempre con un punto e virgola e i blocchi di dichiarazione sono circondati da parentesi graffe.



# Cheatsheet

html → <https://htmlcheatsheet.com/>

css → <https://htmlcheatsheet.com/css/>

Selettori

# Selettori

- Semplici → selezionare gli elementi in base a nome, ID , classe oppure tutti
- Combinatori → seleziona gli elementi in base a una relazione specifica tra loro
- Pseudo-classi → selezionare gli elementi in base a un determinato stato

# Selettori

- Pseudo-elementi → seleziona e disegna una parte di un elemento
- Attributi → selezionare gli elementi in base a un attributo o un valore di attributo


# Semplici

Tag



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

Id



```
#para1 {  
  text-align: center;  
  color: red;  
}
```

Class



```
.center {  
  text-align: center;  
  color: red;  
}
```

# Combinatori

## Discendente



```
div p {  
  background-color: yellow;  
}
```

## Figlio



```
div > p {  
  background-color: yellow;  
}
```

## Fratello Adiacente



```
div + p {  
  background-color: yellow;  
}
```

## Fratello Generico



```
div ~ p {  
  background-color: yellow;  
}
```

# Combinatori

Un combinatore è qualcosa che spiega la relazione tra i selettori.

Un selettore CSS può contenere più di un semplice selettore. Tra i selettori semplici, possiamo includere un combinatore.

Esistono quattro diversi combinatori nei CSS:

**selettore discendente** (spazio): Il selettore discendente corrisponde a tutti gli elementi che sono discendenti di un elemento specificato.

**selettore figlio** (>): Il selettore figlio seleziona tutti gli elementi che sono i figli di un elemento specificato.

# Combinatori

**selettore fratello adiacente** (+): Il selettore fratello adiacente seleziona tutti gli elementi che sono fratelli adiacenti di un elemento specificato. Gli elementi del fratello devono avere lo stesso elemento genitore e "adiacente" cioè "immediatamente successivo".

**selettore generale di pari livello** (~): Il selettore di fratelli generali seleziona tutti gli elementi che sono fratelli di un elemento specificato.

Esercizio divertente: <https://flukeout.github.io/>



# Pseudo-classe

Una pseudo-classe viene utilizzata per definire uno stato speciale di un elemento.

Ad esempio, può essere utilizzato per:

- Cambiare lo stile di un elemento quando un utente si sposta su di esso
- Cambiare lo stile di un collegamento se è stato visitato o meno
- Colora un elemento quando diventa attivo



```
selector:pseudo-class {  
  property:value;  
}
```



```
/* unvisited link */  
a:link {  
  color: #FF0000;  
}  
  
/* visited link */  
a:visited {  
  color: #00FF00;  
}  
  
/* mouse over link */  
a:hover {  
  color: #FF00FF;  
}  
  
/* selected link */  
a:active {  
  color: #0000FF;  
}
```

# Pseudo-elemento

```
selector::pseudo-element {  
  property:value;  
}
```

```
p::first-line {  
  color: #ff0000;  
  font-variant: small-caps;  
}
```

Uno pseudo-elemento CSS viene utilizzato per dare uno stile alle parti specificate di un elemento.

Ad esempio, può essere utilizzato per:

Cambiare lo stile della prima lettera o riga di un elemento

Inserisci il contenuto prima o dopo il contenuto di un altro elemento

<https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-elements>

# Attributi



```
a[target] {  
  background-color: yellow;  
}
```

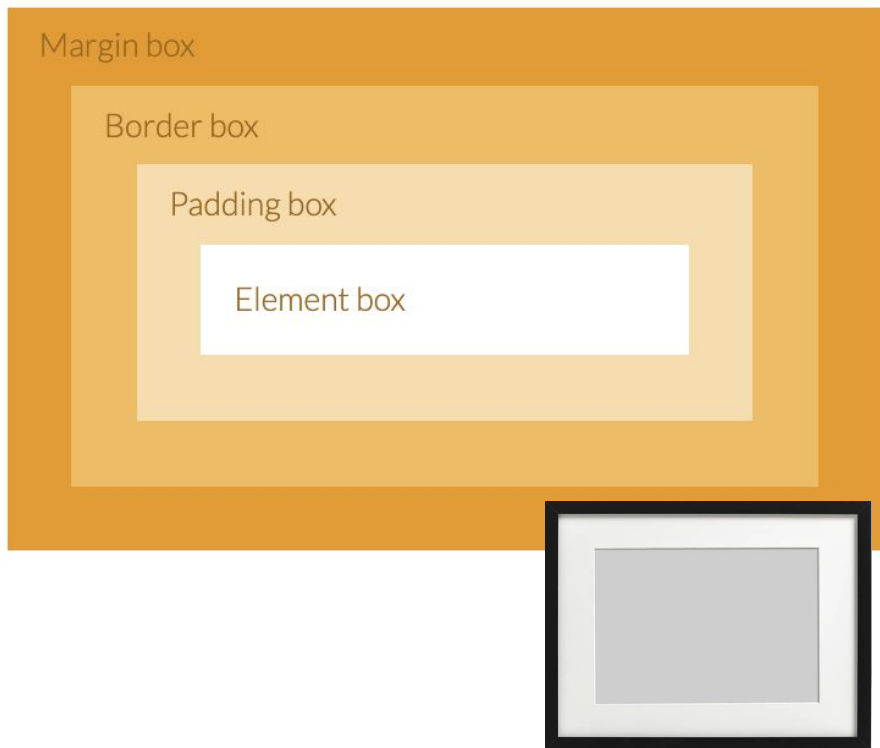


```
input[type="text"] {  
  width: 150px;  
  display: block;  
  margin-bottom: 10px;  
  background-color: yellow;  
}
```

Il selettore [attributo] viene utilizzato per selezionare elementi con un attributo specificato.

# Box model

# Margin e padding



Le proprietà **margin** vengono utilizzate per creare spazio attorno agli elementi, al di fuori di qualsiasi bordo definito. Esistono proprietà per impostare il margine per ciascun lato di un elemento (in alto, a destra, in basso e a sinistra). Le proprietà di **padding** CSS sono utilizzate per generare spazio attorno al contenuto di un elemento, all'interno di qualsiasi bordo definito. Esistono proprietà per l'impostazione dell'imbottitura per ciascun lato di un elemento (in alto, a destra, in basso e a sinistra).

Testo

# Text: font-family

Serif

*Sans-Serif*

Abc

Abc

Sancaole **Muller** *Keepsake*

*Praise* **DARWIN** Chic Chalk

*\*Lingvine* **Magallanes**

Moderna **TRUE NORTH** 

*Merry Melody* Antartida

**Magica** Keswick *\*Garden*

**PF Adamant** *Viento*

# Text: font-size

La proprietà `size-font` imposta la dimensione del testo. Essere in grado di gestire le dimensioni del testo è importante nel web design. Tuttavia, non è necessario utilizzare le regolazioni della dimensione del carattere per rendere i paragrafi simili a titoli o titoli come paragrafi. Usa sempre i tag HTML corretti, come `<h1>` - `<h6>` per le intestazioni e `<p>` per i paragrafi. Il valore della dimensione del carattere può essere una dimensione assoluta o relativa.

6 pt  
8 pt  
9 pt  
10 pt  
11 pt  
12 pt  
14 pt  
18 pt  
24 pt  
30 pt  
36 pt  
48 pt  
60 pt  
72 pt  
84 pt



# Text: font-weight

100 Thin

200 Light

300 Book

400 Regular

500 Medium

600 DemiBold

700 Bold

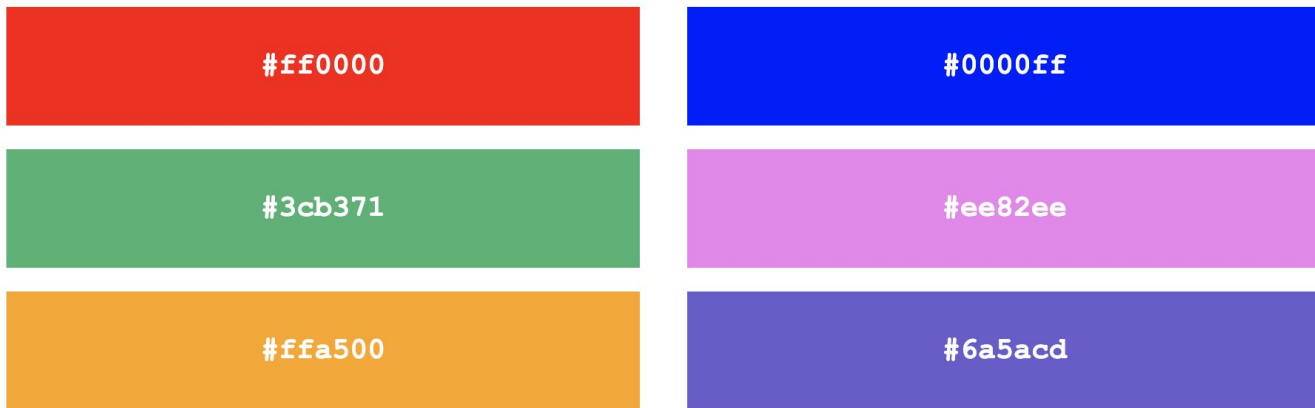
800 ExtraBold

900 Heavy



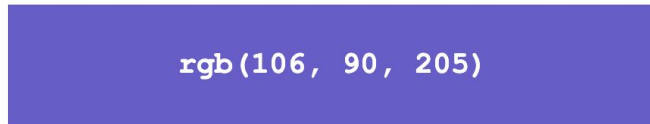
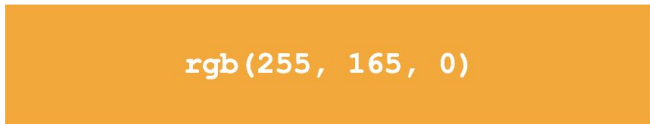
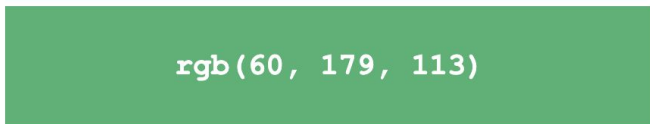
Colori

# Colors HEX



Un colore può essere specificato usando un valore esadecimale nel modulo: #RRGGBB. Dove rr (rosso), gg (verde) e bb (blu) sono valori esadecimali tra 00 e ff (uguale al decimale 0-255).

# Colors RGB



In CSS, i colori possono anche essere specificati usando valori RGB, valori HEX, valori HSL, valori RGBA e valori HSLA. I valori di colore RGBA sono un'estensione dei valori di colore RGB con un canale alfa, che specifica l'opacità di un colore e HSLA stessa cosa ma per HSL.

# Colors HSL

<code>hsl(0, 100%, 50%)</code>	<code>hsl(240, 100%, 50%)</code>
<code>hsl(147, 50%, 47%)</code>	<code>hsl(300, 76%, 72%)</code>
<code>hsl(39, 100%, 50%)</code>	<code>hsl(248, 53%, 58%)</code>

Un colore può essere specificato usando tonalità, saturazione e luminosità (HSL) nel modulo: `hsl(tonalità, saturazione, luminosità)`. La tonalità è un grado sulla ruota dei colori da 0 a 360. 0 è rosso, 120 è verde e 240 è blu. La saturazione è un valore percentuale, 0% indica una tonalità di grigio e il 100% è il colore pieno. Anche la luminosità è una percentuale, lo 0% è nero, il 50% non è né chiaro né scuro, il 100% è bianco

Unità di misura

# Unità di misura

## Absolute

Centimeters (cm)

Millimeters (mm)

Inches (in) (1in = 96px = 2.54cm)

Pixels (px) (1px = 1/96th of 1in)

Points (pt) (1pt = 1/72 of 1in)

Picas (pc) (1pc = 12 pt)

## Relative

Percentages (%)

Font-sizes (em & rem)

Character-sizes (ex & ch)

Viewport Dimensions (vw & vh)

Viewport Max (vmax)

Viewport Min (vmin)

# Unità di misura

Esistono molte unità specifiche della proprietà per i valori utilizzati nei CSS, ma ci sono alcune unità generali utilizzate da un numero di proprietà e vale la pena familiarizzare con queste prima di continuare.

Le unità di lunghezza assoluta sono fisse e una lunghezza espressa in una di queste apparirà esattamente come quella dimensione.

Le unità di lunghezza assoluta non sono consigliate per l'uso sullo schermo, poiché le dimensioni dello schermo variano molto. Tuttavia, possono essere utilizzati se è noto il supporto di output, ad esempio per il layout di stampa.



# Unità di misura

% → Rispetto all'elemento genitore

em → Relativo alla dimensione del carattere dell'elemento (2em significa 2 volte la dimensione del carattere corrente)

rem → Relativo alla dimensione del carattere dell'elemento radice

ex → Relativo all'altezza x del carattere corrente (usato raramente)

ch → Rispetto alla larghezza di "0" (zero)

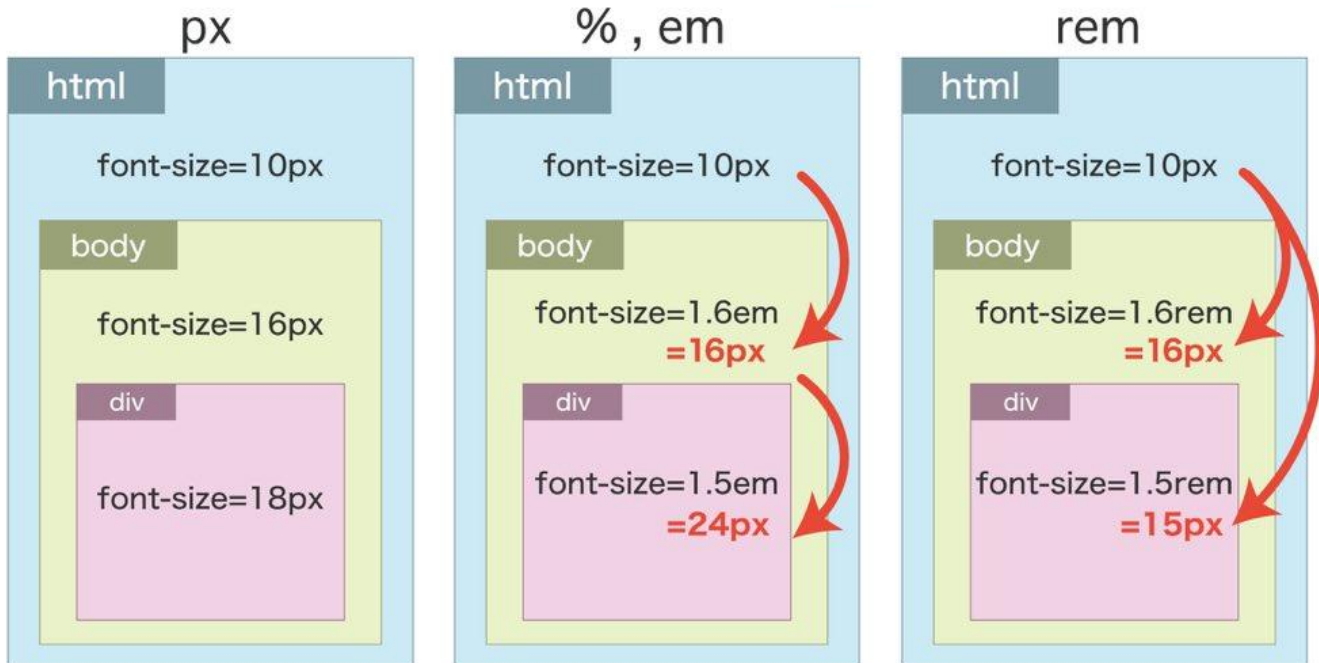
vw → rispetto all'1% della larghezza della finestra \*

vh → rispetto all'1% dell'altezza della finestra \*

vmax → Relativo all'1% della dimensione \* più grande del viewport

vmin → Relativo all'1% della dimensione \* più piccola del viewport

# Unità di misura



# Esercitazioni

[https://www.w3schools.com/html/html\\_exercises.asp](https://www.w3schools.com/html/html_exercises.asp)

<https://www.w3schools.com/quiztest/quiztest.asp?qtest=HTML>

<https://www.w3schools.com/quiztest/quiztest.asp?qtest=CSS>

# Regole e proprietà per rendere un sito responsive

# Media rule



```
@media not|only mediatype and (mediafeature and|or|not mediafeature) {  
  CSS-Code;  
}
```

# Media rule

La regola @media viene utilizzata nelle query multimediali per applicare stili diversi per diversi tipi / dispositivi multimediali.

Le query multimediali possono essere utilizzate per controllare molte cose, come:

- larghezza e altezza del viewport

- larghezza e altezza del dispositivo

- orientamento (il tablet / telefono è in modalità orizzontale o verticale?)

- risoluzione

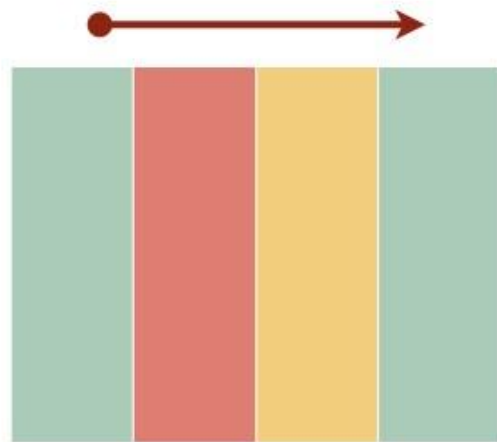
# Media rule

- 320px — 480px: Mobile devices
- 481px — 768px: iPads, Tablets
- 769px — 1024px: Small screens, laptops
- 1025px — 1200px: Desktops, large screens
- 1201px and more — Extra large screens, TV

```
1 @media screen and (max-width: 480px) {  
2   .text {  
3     font-size: 16px;  
4   }  
5 }
```

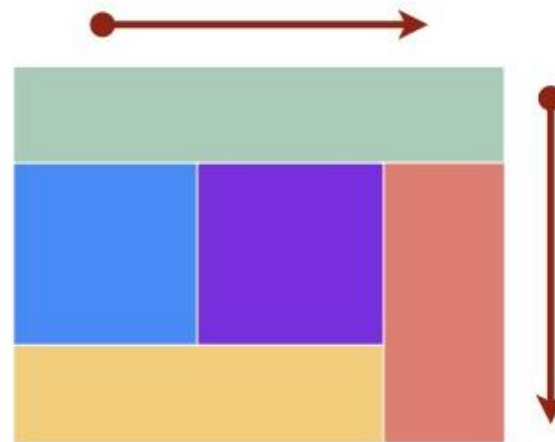
<https://developer.mozilla.org/en-US/docs/Web/CSS/@media>

# Flexbox vs Grid



**Flexbox**  
One Dimensions

VS



**CSS Grids**  
Two Dimensions

<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>  
<https://css-tricks.com/snippets/css/complete-guide-grid/>



# Perché esistono più regole di flex?

`display: box;` → versione 2009

`display: flexbox;` → versione 2011

`display: flex;` → versione attuale

# Esercitazioni

- <https://cssgridgarden.com/#it>
- <https://flexboxfroggy.com/#it>

Link utili:

<https://cssgrid-generator.netlify.app/>

# Stiamo scrivendo del buon codice?

Per controllare se quello che stiamo scrivendo è corretto e segue gli standard possiamo <https://validator.w3.org/> il nostro HTML e il nostro CSS

Validatore HTML → <https://validator.w3.org/>

Validatore CSS → <https://jigsaw.w3.org/css-validator/>

Link utili:

- [https://developer.mozilla.org/en-US/docs/MDN/Writing\\_guidelines/Writing\\_style\\_guide/Code\\_style\\_guide/CSS](https://developer.mozilla.org/en-US/docs/MDN/Writing_guidelines/Writing_style_guide/Code_style_guide/CSS)
- [https://developer.mozilla.org/en-US/docs/Learn/CSS/Building\\_blocks/Organizing](https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Organizing)

# Estensioni utili

<https://marketplace.visualstudio.com/items?itemName=esbenp.prettier-vscode>

# Alcuni approfondimenti

# CSS class naming - BEM

BEM è un metodo di nominare le classi CSS per applicare gli stili all'HTML, quindi non ci sono tag HTML o selettori ID. I nomi delle classi BEM possono essere costituiti da 3 parti  
[blocco]\_\_[elemento]--[modificatore]

Diagram illustrating the BEM naming convention for the class `.product-card__like-button--liked`:

- `product-card`: Block (indicated by a bracket above)
- `__`: 2 underscores (indicated by a bracket below)
- `like-button`: Element (indicated by a bracket above)
- `--`: 2 dashes (indicated by a bracket below)
- `liked`: Modifier (indicated by a bracket above)

The full class name is shown as `.product-card__like-button--liked {` followed by an ellipsis `...` and a closing brace `}`.

The diagram illustrates the application of BEM class names to an HTML structure. On the left, a visual representation of a card is shown with a title, two paragraphs, and a 'READ MORE' button. On the right, the corresponding HTML code is provided, with orange arrows mapping the BEM classes to the elements:

- `<article class="card">` maps to the entire card container.
- `<h1 class="card__title">` maps to the title.
- `<p class="card__text">` maps to the first paragraph.
- `<p class="card__text card__text--secondary">` maps to the second paragraph.
- `<a class="card__button button" href="#">` maps to the 'READ MORE' button.

The HTML code snippet is as follows:

```
<article class="card">
  <h1 class="card__title">Some engaging title</h1>
  <p class="card__text">There are many variations....</p>
  <p class="card__text card__text--secondary">The majority....</p>
  <a class="card__button button" href="#">Read more</a>
</article>
```

# CSS variables

Le proprietà personalizzate (a volte denominate variabili CSS o variabili a cascata) sono entità definite dagli autori del documento CSS che contengono valori specifici da riutilizzare in un documento. Sono impostati utilizzando la notazione della proprietà personalizzata (ad es. `--main-color: black;`) e si accede utilizzando la funzione `var()` (ad es. `color: var(--main-color);`).

```
1 :root {  
2     --primary-color: #333;  
3 }  
4  
5 div {  
6     background-color: var(--primary-color);  
7 }  
8  
9 .test {  
10     color: var(--primary-color);  
11 }
```

[https://developer.mozilla.org/en-US/docs/Web/CSS/--\\*](https://developer.mozilla.org/en-US/docs/Web/CSS/--*)

# Pre e post processori CSS

I preprocessori CSS sono linguaggi di scripting che estendono le capacità predefinite dei CSS. Ci consentono di utilizzare la logica nel nostro codice CSS, come variabili, annidamento, ereditarietà, mixin, funzioni e operazioni matematiche.

<https://www.freecodecamp.org/news/css-preprocessors/#:~:text=CSS%20Preprocessors%20compile%20the%20code,preprocessor%20were%20not%20in%20place.>

<https://sherocommerce.com/what-is-a-css-preprocessors-why-use-them/>



# CSS Framework

Un framework CSS fornisce all'utente un foglio di stile CSS completamente funzionale, consentendo loro di creare una pagina Web semplicemente codificando l'HTML con classi, struttura e ID appropriati. Le classi per le funzionalità di siti Web popolari come il piè di pagina, il dispositivo di scorrimento, la barra di navigazione, il menu dell'hamburger, i layout basati su colonne e così via sono già inclusi nel framework.

<https://css-tricks.com/what-are-the-benefits-of-using-a-css-framework/>

# Animazioni

[https://developer.mozilla.org/en-US/docs/Web/CSS/CSS\\_Animations/Using\\_CSS\\_animations](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Animations/Using_CSS_animations)

<https://css-tricks.com/almanac/properties/a/animation/>

# Come organizzare un progetto?

[https://developer.mozilla.org/en-US/docs/Learn/Getting started with the web/Dealing with files](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/Dealing_with_files)

<https://medium.com/creative-technology-concepts-code/how-to-structure-your-website-files-and-folders-d8521e21c874>