



# Esempi di CSS

*Programmazione*

**CSS (Cascading Style Sheets) è il linguaggio utilizzato per definire lo stile e la presentazione delle pagine web.**

**Permette di controllare colori, spaziature, dimensioni, layout, animazioni e qualsiasi aspetto visivo degli elementi HTML.**

**Grazie alla sua struttura a regole, ogni selettore può essere personalizzato tramite proprietà e valori, facilitando la gestione grafica di un sito web rispetto all'uso del solo HTML.**



**Uno dei punti di forza del CSS è il concetto di “cascata”: quando più regole si applicano allo stesso elemento, il browser decide quale utilizzare in base alla specificità, all’ordine delle dichiarazioni e allo stile inline.**

**Questo rende possibile creare fogli di stile modulari, mantenibili e facilmente estendibili, fondamentali per progetti moderni che richiedono coerenza visiva e aggiornabilità nel tempo.**



## Caratteristiche principali del CSS

- **1. Separazione tra struttura e presentazione**

**Il contenuto viene gestito dall'HTML, mentre il CSS si occupa dello stile. Questo favorisce la manutenzione, riduce la duplicazione del codice e migliora l'accessibilità.**

- **2. Sistema a cascata e specificità**

**Quando più regole colpiscono lo stesso elemento, il browser applica quella più specifica o più recente. Questo meccanismo definisce l'ordine di priorità degli stili.**



## Caratteristiche principali del CSS

- **3. Ereditarietà delle proprietà**

**Alcune proprietà, come font o colore del testo, si propagano automaticamente agli elementi figli, riducendo la necessità di riscrivere le regole più volte.**

- **4. Selettori flessibili e potenti**

**Il CSS offre selettori semplici (tag, classi, ID) e complessi (discendenti, attributi, pseudo-classi e pseudo-elementi) per controllare in modo preciso quali elementi stilizzare.**



## Caratteristiche principali del CSS

- **5. Layout moderni: Flexbox e Grid**

**Questi due sistemi consentono di creare layout complessi, responsivi e dinamici senza ricorrere a hack o strutture HTML complicate.**

- **6. Responsive design e media query**

**Permettono di adattare lo stile a schermi diversi (smartphone, tablet, desktop) modificando dimensioni, layout e visibilità degli elementi.**



## Caratteristiche principali del CSS

- **7. Transizioni e animazioni**

**Il CSS consente di creare effetti dinamici fluidi, come fade-in, slide, hover animati, senza usare JavaScript per i movimenti più semplici.**

- **8. Variabili CSS (custom properties)**

**Consentono di definire valori riutilizzabili (come colori e margini), facilitando la gestione di temi e design system.**



## **Caratteristiche principali del CSS**

- **9. Compatibilità cross-browser**

**Anche se gli standard evolvono, il CSS include meccanismi per mantenere comportamenti coerenti tra browser diversi tramite prefix o fallback.**

- **10. Modularità e riusabilità**

**È possibile dividere gli stili in più file, utilizzare classi riutilizzabili e creare sistemi di design scalabili e professionali.**





## *Separazione tra struttura e presentazione*

**L'HTML descrive il contenuto, mentre il CSS definisce lo stile. Questo evita duplicazioni e rende il progetto più manutenibile.**

**Nel codice sotto, l'HTML contiene solo il testo; il CSS decide il colore e la dimensione del paragrafo.**

### **HTML**

1. `<p class="testo">Questo è un paragrafo.</p>`

### **CSS**

```
1..testo {  
2.   color: blue;  
3.   font-size: 20px;  
4.}
```



**Il browser carica l'HTML, applica la classe .testo e mostra il paragrafo in blu e più grande.**

## ***Sistema a Cascata e Specificità***

**Se più regole si applicano allo stesso elemento, vince quella più specifica.**

**Tre regole si applicano allo stesso <p>, ma ID è più specifico della classe e del tag.**

### **HTML**

1. <p id="principale" class="paragrafo">Testo</p>

### **CSS**

1. p { color: black; }

2. .paragrafo { color: green; }

3. #principale { color: red; }

**Il testo risulta rosso, perché l'ID ha la specificità più alta.**



## ***Ereditarietà delle proprietà***

**Molte proprietà (come il font) si trasmettono agli elementi figli.**

**Il contenitore imposta un font, i paragrafi lo ereditano automaticamente.**

### **HTML**

1. `<div class="container">`
2.   `<p>Primo paragrafo</p>`
3.   `<p>Secondo paragrafo</p>`
4. `</div>`

### **CSS**

1. `.container {`
2.   `font-family: Arial, sans-serif;`
3. `}`



**Entrambi i paragrafi usano lo stesso font definito nel contenitore.**

## ***Selettori flessibili (base + avanzati)***

**I selettori permettono di colpire elementi specifici o basati sul loro stato.**

**Usiamo una classe, un selettore discendente e una pseudo-classe :hover.**

### **HTML**

1. `<div class="card">`
2. `<p>Testo dentro la card</p>`
3. `</div>`

### **CSS**

1. `.card p {`
2. `color: #333;`
3. `}`
- 4.
5. `.card:hover {`
6. `background-color: #f0f0f0;`
7. `}`



**Entrambi i paragrafi usano lo stesso font definito nel contenitore.**

## ***Layout moderni: Flexbox***

**Flexbox allinea gli elementi facilmente anche in modo responsive.**

**Gli elementi nella navbar si distribuiscono ai lati con space-between.**

### **HTML**

1. `<div class="navbar">`
2.   `<div>Logo</div>`
3.   `<div>Menu</div>`
4. `</div>`

### **CSS**

1. `.navbar {`
2.   `display: flex;`
3.   `justify-content: space-between;`
4. `}`



**Gli elementi si dispongono agli estremi della barra, senza usare float o hack**

## ***Responsive Design con Media Query***

**Le media query cambiano lo stile in base alla dimensione dello schermo.**

**Se lo schermo è piccolo (max 600px), il testo diventa più grande per migliorare la lettura.**

### **CSS**

```
1.p {  
2.  font-size: 16px;  
3.}  
4.  
5.@media (max-width: 600px) {  
6.  p {  
7.    font-size: 20px;  
8.  }  
9.}
```



**Su smartphone il testo appare più grande rispetto alla versione desktop**

## ***Transizioni e Animazioni***

**Si possono animare proprietà CSS senza usare JavaScript.**

**L'elemento aumenta lo zoom quando ci passi sopra, animato da una transizione.**

### **CSS**

```
1. box {  
2.   transition: transform 0.3s;  
3. }  
4.  
5. box:hover {  
6.   transform: scale(1.1);  
7. }
```

### **HTML**

```
1. <div class="box">Hover me</div>
```



**Quando l'utente passa il mouse, la box cresce in modo fluido.**

## ***Variabili CSS (Custom Properties)***

**Le variabili permettono di definire colori e dimensioni riutilizzabili.  
Definiamo una variabile `--main-color` e la riutilizziamo in più elementi.**

### **CSS**

```
1.:root {  
2.  --main-color: #0066ff;  
3.}  
4.  
5.button {  
6.  background-color: var(--main-color);  
7.  color: white;  
8.}
```

**Cambiando il valore di `--main-color`, tutti i pulsanti cambiano colore automaticamente.**





## ***Variabili CSS (Custom Properties)***

**Le variabili permettono di definire colori e dimensioni riutilizzabili.  
Definiamo una variabile `--main-color` e la riutilizziamo in più elementi.**

### **CSS**

```
1.:root {  
2.  --main-color: #0066ff;  
3.}  
4.  
5.button {  
6.  background-color: var(--main-color);  
7.  color: white;  
8.}
```

**Cambiando il valore di `--main-color`, tutti i pulsanti cambiano colore automaticamente.**



**Buon Davante a tutti**

