

Lezione 2

CSS, Bootstrap e Tailwind CSS

Simone Colli

27 ottobre 2025

- 1 Contenuto della lezione
- 2 CSS - Cascading Style Sheets
- 3 Bootstrap
- 4 Tailwind CSS
- 5 Confronto tra approcci

- CSS: sintassi base, selettori, box model, flexbox.
- Bootstrap: grid system, componenti principali, utilities.
- Tailwind CSS: utility-first approach, classi comuni, customizzazione.
- Confronto pratico tra i tre approcci.

Cos'è CSS?

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

Separazione delle responsabilità

- **HTML**: struttura e contenuto semantico
- **CSS**: presentazione e layout visivo
- **JavaScript**: comportamento e interattività (prossima lezione)

Vantaggi:

- Riutilizzo degli stili su più pagine
- Manutenibilità del codice
- Migliore performance (caching CSS)
- Responsive design

Sintassi CSS base

```
/* Selettore { propriet: valore; } */

h1 {
  color: #333;
  font-size: 2rem;
  margin-bottom: 1rem;
}

.container {
  max-width: 1200px;
  margin: 0 auto;
  padding: 20px;
}

#header {
  background-color: #f8f9fa;
  border-bottom: 1px solid #dee2e6;
}
```

Selettori di base

```
/* Elemento */
p { color: black; }

/* Classe */
.button { padding: 10px; }

/* ID */
#navbar { position: fixed; }

/* Attributo */
input[type="text"] {
    border: 1px solid #ccc;
}
```

Selettori combinati

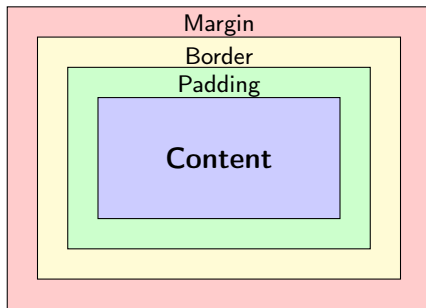
```
/* Discendente */
div p { margin: 0; }

/* Figlio diretto */
ul > li { list-style: none; }

/* Adiacente */
h1 + p { font-size: 1.2em; }

/* Pseudo-classi */
a:hover { color: blue; }
button:active {
    transform: scale(0.95);
}
```

Box Model



```
.box {  
  width: 300px;           /* Larghezza content */  
  padding: 20px;          /* Spazio interno */  
  border: 2px solid #000; /* Bordo */  
  margin: 10px;           /* Spazio esterno */  
}  
/* Larghezza totale: 300 + 40 (padding) + 4 (border) + 20 (margin) = 364px */
```

Flexbox - Layout flessibile

Flexbox permette di creare layout flessibili e responsive con facilità.

```
.container {  
  display: flex;  
  justify-content: space-between; /* allineamento orizzontale */  
  align-items: center;           /* allineamento verticale */  
  gap: 20px;                    /* spazio tra elementi */  
}  
  
.item {  
  flex: 1;                      /* crescita proporzionale */  
  min-width: 0;                 /* per gestire overflow */  
}
```

Alcune proprietà

- flex-direction: row | column | row-reverse | column-reverse
- justify-content: flex-start | center | space-between | space-around
- align-items: flex-start | center | flex-end | stretch

Esempio pratico CSS

```
/* Stile per Task Manager */
.task-card {
  background: white;
  border: 1px solid #e0e0e0;
  border-radius: 8px;
  padding: 16px;
  margin-bottom: 12px;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
  transition: transform 0.2s, box-shadow 0.2s;
}

.task-card:hover {
  transform: translateY(-2px);
  box-shadow: 0 4px 8px rgba(0,0,0,0.15);
}
```

Esempio pratico CSS

```
.task-title {  
  font-size: 1.25rem;  
  font-weight: 600;  
  color: #333;  
  margin-bottom: 8px;  
}  
  
.task-priority-high {  
  border-left: 4px solid #e74c3c;  
}
```

Cos'è Bootstrap?

Bootstrap è il framework CSS più popolare per sviluppare interfacce responsive e mobile-first.

Caratteristiche principali

- **Grid System:** sistema a 12 colonne flessibile
- **Componenti:** navbar, cards, modals, forms, buttons...
- **Utilities:** classi predefinite per spacing, colori, typography
- **JavaScript:** componenti interattivi (dropdown, carousel, modal...)
- **Responsive:** breakpoints per diversi dispositivi

Installazione Bootstrap

Per usare bootstrap si può utilizzare due modi principali:

- **Via CDN:** includere i link CSS e JS direttamente nell'HTML.
- **Via npm:** installare tramite npm/yarn per progetti più complessi.

Molto importante durante il tutorato e lo sviluppo del progetto per il corso **non si utilizzerà npm**.

Nota

Per usare i componenti JavaScript di Bootstrap, includere sempre il bundle JS.

Grid System di Bootstrap

Sistema a **12 colonne** con breakpoints responsive:

```
<div class="container">
  <div class="row">
    <div class="col-md-8">
      <!-- Contenuto principale: 8 colonne su medium+ -->
    </div>
    <div class="col-md-4">
      <!-- Sidebar: 4 colonne su medium+ -->
    </div>
  </div>

  <div class="row">
    <div class="col-sm-6 col-lg-3">
      <!-- 6 colonne su small, 3 su large -->
    </div>
    <!-- Ripeti per altre colonne -->
  </div>
</div>
```

Breakpoints: xs (<576px), sm (≥576px), md (≥768px), lg (≥992px), xl (≥1200px)

Cos'è Tailwind CSS?

Tailwind CSS è un framework CSS **utility-first** che fornisce classi atomiche per costruire interfacce custom.

Filosofia Utility-First

Invece di componenti predefiniti, Tailwind offre classi utility di basso livello per costruire design personalizzati direttamente nell'HTML.

Vantaggi:

- **Rapidità:** styling inline senza scrivere CSS custom
- **Consistenza:** design system integrato (colori, spacing, typography)
- **Customizzazione:** configurabile tramite `tailwind.config.js`
- **Tree-shaking:** include solo le classi utilizzate (file CSS minimo)
- **Responsive:** modificatori per ogni breakpoint

Per usare Tailwind CSS, si può scegliere tra due metodi principali:

- **Via CDN:** includere lo script CDN per prototipazione rapida.
- **Setup completo con build process:** raccomandato per progetti reali.

Molto importante durante il tutorato e lo sviluppo del progetto per il corso **non si utilizzerà npm**.

Classi Utility Tailwind - Esempi

```
<!-- Spacing: m=margin, p=padding, numeri in multipli di 0.25rem -->
<div class="mt-4 mb-6 px-3 py-2">

<!-- Colori: bg-{color}-{shade} -->
<div class="bg-blue-500 text-white">
<p class="text-gray-700">Testo grigio</p>

<!-- Layout & Flexbox -->
<div class="flex justify-between items-center gap-4">
<div class="grid grid-cols-3 gap-6">

<!-- Dimensioni -->
<div class="w-64 h-32"> <!-- width: 16rem, height: 8rem -->
<img class="w-full h-auto">

<!-- Bordi & Ombre -->
<div class="border border-gray-300 rounded-lg shadow-md">

<!-- Responsive: prefisso md:, lg:, xl: -->
<div class="w-full md:w-1/2 lg:w-1/3">
```


Responsive Design con Tailwind

Tailwind usa modificatori per i breakpoints: sm:, md:, lg:, xl:, 2xl:

```
<div class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-4">
  <!-- 1 colonna su mobile, 2 su tablet, 3 su desktop -->
</div>

<nav class="flex flex-col md:flex-row gap-4">
  <!-- Menu verticale su mobile, orizzontale su tablet+ -->
</nav>

<img class="w-full md:w-1/2 lg:w-1/3">
  <!-- 100% larghezza su mobile, 50% su tablet, 33% su desktop -->

<h1 class="text-2xl md:text-4xl lg:text-5xl">
  <!-- Font size progressivo -->
</h1>

<div class="hidden lg:block">
  <!-- Visibile solo da desktop in su -->
</div>
```

CSS vs Bootstrap vs Tailwind

Aspetto	CSS Puro	Bootstrap	Tailwind
Curva di apprendimento	Medio-alta (conoscenza completa)	Bassa (documentazione chiara)	Media da riconfermare
Velocità sviluppo	Più lenta	Veloce con componenti	Molto pratica
Personalizzazione	Massima libertà	Buona (variables SASS)	Ottima
Dimensione file	Minima (solo necessario)	20KB (minified)	10KB
Design	Completamente custom	Look "Bootstrap"	Completo custom
Manutenibilità	Ottima con BEM/-metodologie	Buona	Buona verboso

Stesso componente - tre approcci

CSS Puro

```
<div class="task-card">
  <h3 class="task-title">
    Task
  </h3>
  <p class="task-desc">
    Descrizione
  </p>
</div>

<style>
.task-card {
  background: white;
  padding: 1rem;
  border-radius: 8px;
  box-shadow: 0 2px 4px
    rgba(0,0,0,0.1);
}
.task-title {
  font-size: 1.25rem;
  font-weight: 600;
}
</style>
```

Bootstrap

```
<div class="card">
  <div class="card-body">
    <h5 class="card-title">
      Task
    </h5>
    <p class="card-text">
      Descrizione
    </p>
  </div>
</div>
```

Tailwind

```
<div class="bg-white">
  <h3 class="text-xl
    font-semibold">
    Task
  </h3>
  <p class="text-gray-600">
    Descrizione
  </p>
</div>
```

Quale approccio scegliere?

Quale scegliere?

CSS puro: progetti custom, team con forte competenza CSS

Bootstrap: prototipazione rapida, progetti aziendali standard

Tailwind: massima flessibilità, team che preferisce utility-first