

Modulo 2

Web Tier

CSS - Day 1

Presentato da
Vitale Esca
IBM Client Innovation Center - Italy

20/11/2023

IBM Client Innovation Center
Italy

Agenda Day 1

- 1 Concetti di Base su CSS
- 2 Responsiveness di una pagina web
- 3 Introduzione a Bootstrap CSS
- 4 Pratica

Cosa è CSS e a cosa serve

Il CSS (Cascading Style Sheet) è il linguaggio usato per definire la formattazione di un documento HTML, ad esempio il colore del testo

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      body {
        color: blue;
      }
      h1 {
        color: green;
      }
    </style>
  </head>
  <body>
    <h1>This is heaging</h1>
    <p>
      This is an ordinary paragraph.
      Notice that this text is blue.
      The default text color for a page is defined in the body selector.</p>
    <p>Another paragraph.</p>
  </body>
</html>
```

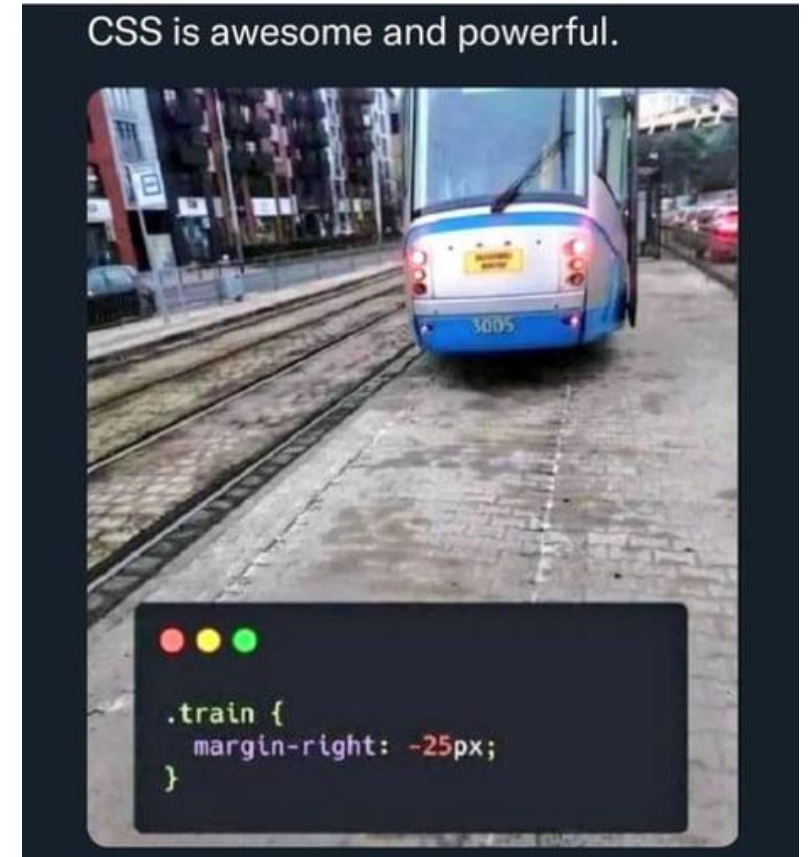
This is heaging

This is an ordinary paragraph. Notice that this text is blue. The default text color for a page is defined in the body selector.

Another paragraph.

Cos'è e a cosa serve un foglio di stile

Un foglio di stile permette di definire lo stile da assegnare ad un elemento singolo nella pagina tramite il suo **id**, ad una **classe** di elementi o ad un'intera **tipologia di elementi**, in un file separato, rendendo il codice riutilizzabile su altre pagine HTML



Integrare un foglio di stile nella pagina HTML

Per integrare un foglio di stile nella pagina HTML, utilizziamo il tag `<link>` utilizzando la keyword **stylesheet** per l'attributo *rel* e il path relativo del foglio di stile nell'attributo *href*.

E' consigliato includere il file css nel campo `<head>`

```
<!DOCTYPE html>
<html>

<head>
  <link rel="stylesheet" href="mystyle.css">
</head>

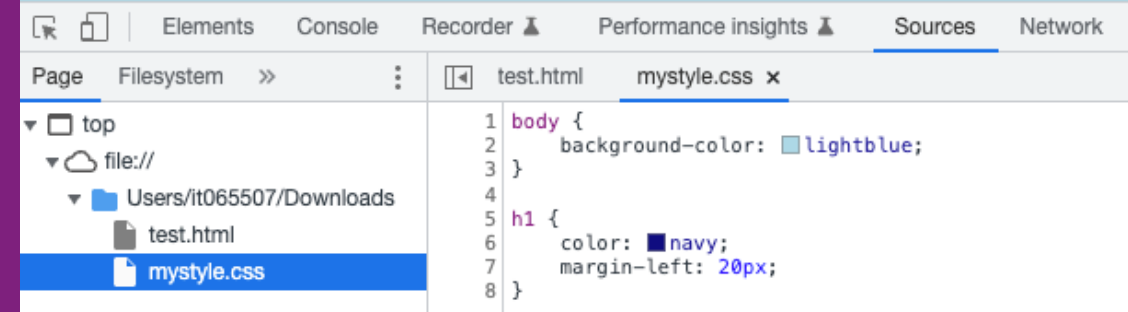
<body>

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

</html>
```

This is a heading

This is a paragraph.



Il selettore elemento in HTML e CSS

Il selettore **elemento** seleziona tutti gli elementi HTML con il nome dell'elemento specificato

Demo of the element selector

My name is Donald.

I live in Duckburg.

My best friend is Mickey.

```
<!DOCTYPE html>
<html>

<head>
  <style>
    p {
      background-color: yellow;
    }
  </style>
</head>

<body>

  <h1>Demo of the element selector</h1>

  <div>
    <p id="firstname">My name is Donald.</p>
    <p id="hometown">I live in Duckburg.</p>
  </div>

  <p>My best friend is Mickey.</p>

</body>

</html>
```

Day 1 – Concetti di Base su CSS

Il selettore *class*

Il selettore **class** seleziona tutti gli elementi HTML con uno specifico attributo di classe, in modo da poter assegnare uno stile comune a tutti gli elementi

```
<!DOCTYPE html>
<html>

<head>
  <style>
    .center {
      text-align: center;
      color: red;
    }
  </style>
</head>

<body>

  <h1 class="center">Red and center-aligned heading</h1>
  <p class="center">Red and center-aligned paragraph.</p>

</body>

</html>
```

Red and center-aligned heading

Red and center-aligned paragraph.

Per approfondimenti: https://www.w3schools.com/cssref/sel_class.asp

Il selettore id in HTML e CSS

Il selettore **id** identifica univocamente uno specifico elemento HTML, in modo da poter assegnare uno stile specifico per quell'elemento. Per selezionare un elemento con uno specifico **id**, bisogna utilizzare il carattere **#** seguito dal nome dell'id dell'elemento

```
<!DOCTYPE html>
<html>

<head>
  <style>
    #para1 {
      text-align: center;
      color: red;
    }
  </style>
</head>

<body>

  <p id="para1">Hello World!</p>
  <p>This paragraph is not affected by the style.</p>

</body>

</html>
```

Hello World!

This paragraph is not affected by the style.

Specificità dei selettori CSS

Molto spesso capita di scrivere regole CSS che vengono sovrascritte da altre.

Questo succede perchè esiste una gerarchia di importanza per i selettori CSS che abbiamo analizzato nelle slide precedenti. Il loro ordine di specificità è il seguente:

1. id
2. class
3. element

Attenzione agli **!important** e allo stile **inline**

Day 1 – Concetti di Base su CSS

Proprietà *margin*

- Con margin si definisce lo spazio attorno all'elemento a partire dal bordo dell'elemento
- Si definisce tramite la keyword `margin-*` per una determinata classe o id
- Con il valore **auto** si centra l'elemento orizzontalmente all'interno del proprio container. Il margine calcolato a destra e sinistra sarà uguale, sottraendo alla larghezza totale quella dell'elemento
- Spesso si usa la sintassi abbreviata, con il quale è possibile definire le proprietà in una sola riga. L'ordine in questo caso segue il senso **orario**:
top - right - bottom - left

```
margin: 25px 50px 75px 100px;
/*
  top margin is 25px
  right margin is 50px
  bottom margin is 75px
  left margin is 100px
*/
```

```
margin: 25px 50px;
/*
  top and bottom margins are 25px
  right and left margins are 50px
*/
```

```
p {
  margin-top: 100px;
  margin-bottom: 100px;
  margin-right: 150px;
  margin-left: 80px;
}
```

Attenzione! Non è obbligatorio definire il margin per tutti i lati, ma anche solo ad esempio quello destro o sinistro!

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  border: 1px solid black;
  margin-top: 100px;
  margin-bottom: 100px;
  margin-right: 150px;
  margin-left: 80px;
  background-color: lightblue;
}
</style>
</head>
<body>

<h2>Using individual margin properties</h2>

<div>This div element has a top margin of 100px,
a right margin of 150px, a bottom margin of
100px, and a left margin of 80px.</div>

</body>
</html>
```

Using individual margin properties

This div element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 80px.

Day 1 – Concetti di Base su CSS

Proprietà *padding*

- Con *padding* si definisce lo spazio attorno all'elemento a partire dal contenuto dell'elemento
- La sintassi prevede la keyword `padding-*` per una determinata classe o id
- Spesso si usa la sintassi abbreviata, con il quale è possibile definire le proprietà in una sola riga. L'ordine in questo caso segue il senso **orario**: *top - right - bottom - left*

```
padding: 25px 50px 75px 100px;
/*
  top padding is 25px
  right padding is 50px
  bottom padding is 75px
  left padding is 100px
*/
```

```
padding: 25px 50px 75px;
/*
  top padding is 25px
  right and left paddings are 50px
  bottom padding is 75px
*/
```

```
<!DOCTYPE html>
<html>
<head>
  <style>
    div {
      border: 1px solid black;
      background-color: lightblue;
      padding-top: 50px;
      padding-right: 30px;
      padding-bottom: 50px;
      padding-left: 80px;
    }
  </style>
</head>
<body>
  <h2>Using individual padding properties</h2>

  <div>
    This div element has a top padding of 50px, a right
    padding of 30px, a bottom padding of 50px, and a left
    padding of 80px.
  </div>
</body>
</html>
```

Attenzione! Non è obbligatorio definire il padding per tutti i lati, ma anche solo ad esempio quello destro o sinistro!

Using individual padding properties

This div element has a top padding of 50px, a right padding of 30px, a bottom padding of 50px, and a left padding of 80px.

Day 1 – Concetti di Base su CSS

Proprietà display

Con `display` si controlla come e se un elemento viene renderizzato dal browser.

Alcuni valori che può assumere `display` sono:

- **block**
 - l'elemento occuperà tutto lo spazio disponibile sulla linea e spinge gli elementi successivi a una nuova linea (di default lo sono i `<div>`, `<p>`, `<h1>`...)
- **inline**
 - l'elemento verrà posizionato sulla linea e non è possibile definire le caratteristiche dell'elemento quali dimensioni, padding, margin, ecc... Di default uno ``
- **inline-block**
 - viene trattato come un block, però senza breakline. Inoltre posso impostare dimensioni, padding e margin
- **None**
 - l'elemento verrà nascosto alla vista

`display: block;`

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      span {
        background:lightblue;
        display:block;
      }
    </style>
  </head>
  <body>
    <div id="myDIV">
      <p>A little demo:</p>
      <p>Here we have place a <span>little blue SPAN</span> element.</p>
    </div>
  </body>
</html>
```

A little demo:

Here we have place a
little blue SPAN
element.

`display: inline-block;`

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      span {
        background:lightblue;
        display:inline-block;
      }
    </style>
  </head>
  <body>
    <div id="myDIV">
      <p>A little demo:</p>
      <p>Here we have place a <span>little blue SPAN</span> element.</p>
    </div>
  </body>
</html>
```

A little demo:

Here we have place a little blue SPAN element.

Per approfondimenti: https://www.w3schools.com/cssref/pr_class_display.asp

Day 1 – Concetti di Base su CSS

Proprietà position

- Con `position` si specifica la posizione di un elemento nel documento HTML

Può assumere i seguenti valori:

- `absolute`
- `fixed`
- `relative`
- `static`

CSS Property:

position:

- ☒ static
- ☐ absolute
- ☐ fixed
- ☐ relative
- ☐ initial

Result:



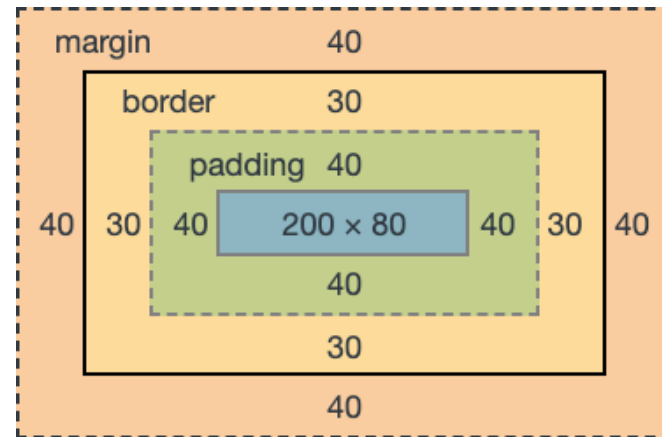
CSS Code:

```
div#myDIV {  
  position:static;  
  width:100px;  
  height:100px;  
  background:red;  
  left:10px;  
  top:100px;  
}
```

Margin o Padding? Differenze...

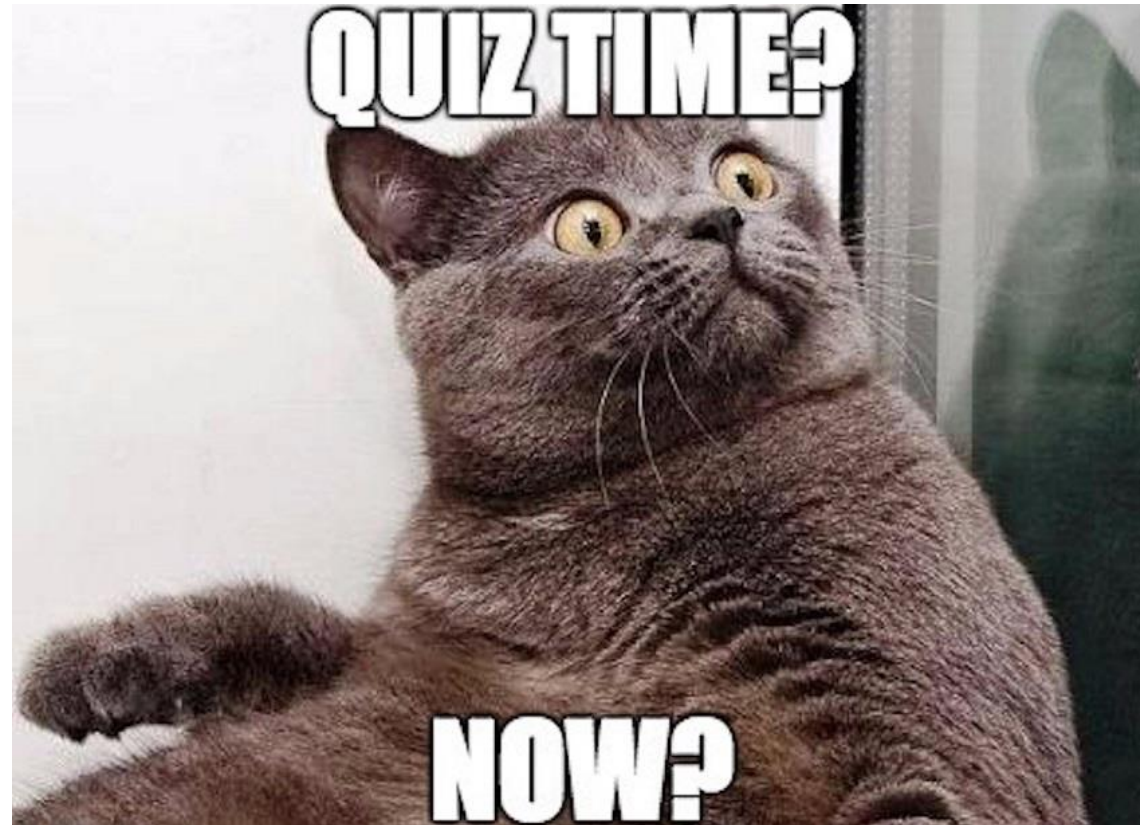
- **margin** è lo spazio attorno al bordo di un elemento
- **margin** controlla lo spazio all'esterno di un elemento

- **padding** è lo spazio tra il bordo di un elemento e il contenuto dell'elemento
- **padding** controlla lo spazio all'interno di un elemento



Per approfondimenti:
<https://blog.hubspot.com/website/css-margin-vs-padding>
<https://www.techgeekbuzz.com/blog/css-margin-vs-padding/>

Day 1 – Quiz



Day 1 – Concetti di Base su CSS

Formattazione del testo

È possibile specificare le caratteristiche da assegnare ad un testo presente nella pagina, come ad esempio:

font-style definisce la formattazione del testo

```
p.normal {  
    font-style: normal;  
}  
  
p.italic {  
    font-style: italic;  
}  
  
p.oblique {  
    font-style: oblique;  
}
```

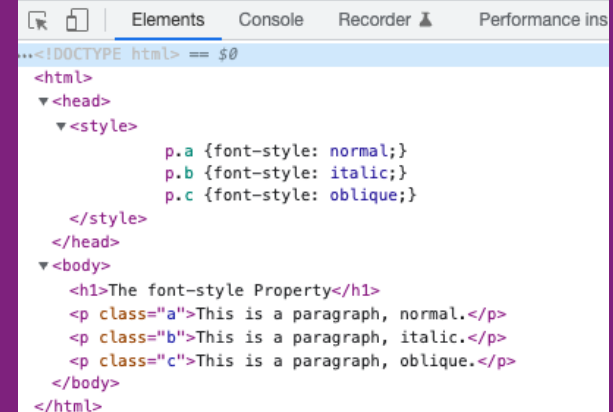
```
<!DOCTYPE html>  
<html>  
<head>  
    <style>  
        p.a {font-style: normal;}  
        p.b {font-style: italic;}  
        p.c {font-style: oblique;}  
    </style>  
</head>  
<body>  
    <h1>The font-style Property</h1>  
    <p class="a">This is a paragraph, normal.  
</p>  
    <p class="b">This is a paragraph, italic.  
</p>  
    <p class="c">This is a paragraph, oblique.  
</p>  
</body>  
</html>
```

The font-style Property

This is a paragraph, normal.

This is a paragraph, italic.

This is a paragraph, oblique.



```
...<!DOCTYPE html> == $0  
<html>  
  <head>  
    <style>  
      p.a {font-style: normal;}  
      p.b {font-style: italic;}  
      p.c {font-style: oblique;}  
    </style>  
  </head>  
  <body>  
    <h1>The font-style Property</h1>  
    <p class="a">This is a paragraph, normal.</p>  
    <p class="b">This is a paragraph, italic.</p>  
    <p class="c">This is a paragraph, oblique.</p>  
  </body>  
</html>
```

Per approfondimenti: https://www.w3schools.com/cssref/pr_font_font-style.asp

Day 1 – Concetti di Base su CSS

Formattazione del testo

font-size definisce la grandezza del testo

```
<h1 style="font-size:10vw">Hello World</h1>
```

```
<!DOCTYPE html>
<html>
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
  <h1 style="font-size:10vw;">Responsive Text</h1>
  <p style="font-size:5vw;">Resize the browser window to see how the text size scales.
</p>
  <p style="font-size:5vw;">Use the "vw" unit when sizing the text. 10vw will set the
size to 10% of the viewport
width.</p>
  <p>Viewport is the browser window size. 1vw = 1% of viewport width. If the viewport
is 50cm wide, 1vw is 0.5cm.</p>
</body>
</html>
```

Responsive Text

Resize the browser window to see how the text size scales.

Use the "vw" unit when sizing the text. 10vw will set the size to 10% of the viewport width.

Viewport is the browser window size. 1vw = 1% of viewport width. If the viewport is 50cm wide, 1vw is 0.5cm.

Day 1 – Concetti di Base su CSS

Formattazione del testo

font-family definisce il font desiderato

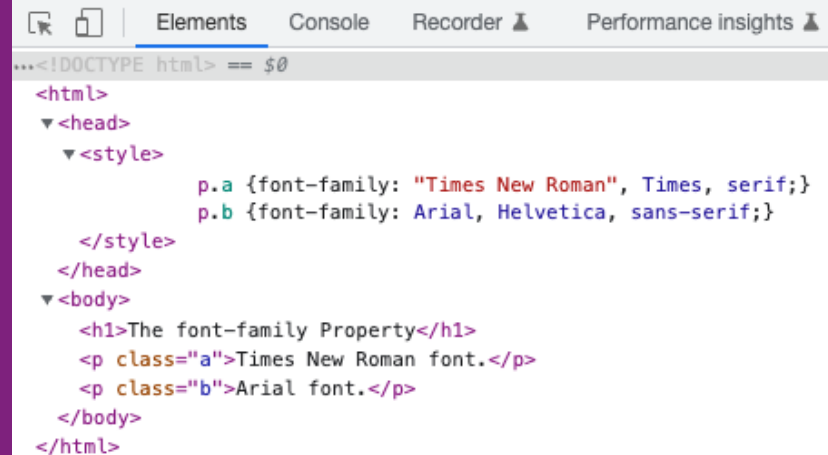
```
p.a {  
  font-family: "Times New Roman", Times, serif;  
}  
p.b {  
  font-family: Arial, Helvetica, sans-serif;  
}
```

```
<!DOCTYPE html>  
<html>  
  <head>  
    <style>  
      p.a {font-family: "Times New Roman", Times, serif;}  
      p.b {font-family: Arial, Helvetica, sans-serif;}  
    </style>  
  </head>  
  <body>  
    <h1>The font-family Property</h1>  
    <p class="a">Times New Roman font.</p>  
    <p class="b">Arial font.</p>  
  </body>  
</html>
```

The font-family Property

Times New Roman font.

Arial font.



```
...<!DOCTYPE html> == $0  
<html>  
  <head>  
    <style>  
      p.a {font-family: "Times New Roman", Times, serif;}  
      p.b {font-family: Arial, Helvetica, sans-serif;}  
    </style>  
  </head>  
  <body>  
    <h1>The font-family Property</h1>  
    <p class="a">Times New Roman font.</p>  
    <p class="b">Arial font.</p>  
  </body>  
</html>
```

Per approfondimenti: https://www.w3schools.com/cssref/pr_font_font-family.asp

Day 1 – Concetti di Base su CSS

L'importanza dello «stile» in un sito web

Attenzione!

Quando si crea una pagina web o una qualsiasi interfaccia, bisogna tenere bene a mente che la **bellezza di una schermata**, data dalla *pulizia* dell'UI e l'oculata scelta di colori e formattazione del testo/pulsanti, incide in modo molto marcato sulla qualità percepita dall'utente finale del vostro software, quindi investiteci sempre una parte del vostro tempo di sviluppo!



“These authors found a significant relationship between web design, perceived quality, and customer trust in e-commerce.”

Per approfondimenti:

[https://www.tandfonline.com/doi/full/10.1080/23311975.2020.1869363#:~:text=2.2.-,Perceived%20website%20quality,1\)%2C%201563%E2%80%93931574](https://www.tandfonline.com/doi/full/10.1080/23311975.2020.1869363#:~:text=2.2.-,Perceived%20website%20quality,1)%2C%201563%E2%80%93931574)

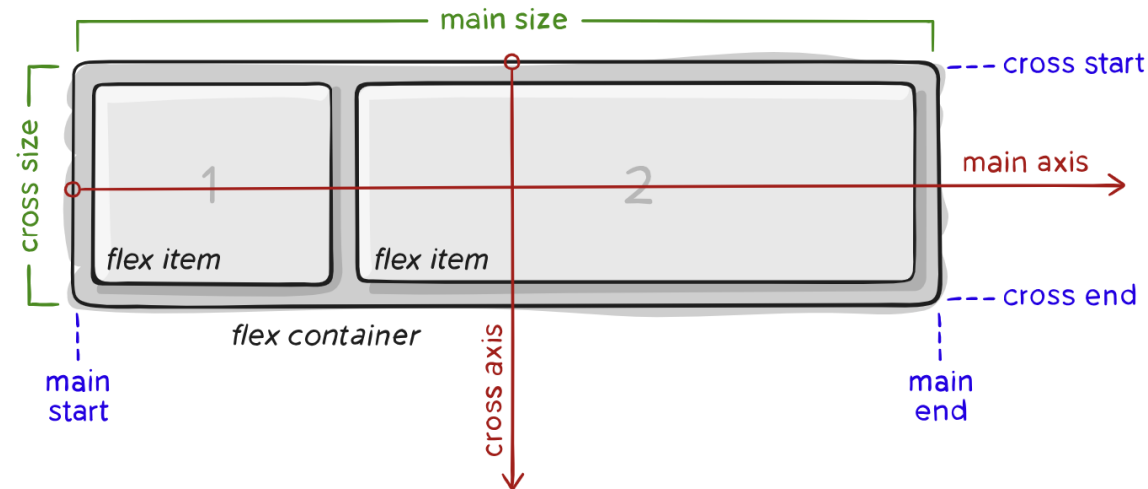
<https://mailchimp.com/resources/how-to-design-a-beautiful-website/>

Day 1 – Concetti di Base su CSS

Flexbox

Il **Flexbox Layout** è un modulo CSS che mira a fornire un modo efficiente per disporre e allineare lo spazio tra gli elementi di un contenitore anche quando la loro dimensione è sconosciuta o dinamica.

Gli elementi sono disposti in maniera flessibile secondo l'asse principale o trasversale di un contenitore.



Per approfondimenti: https://www.w3schools.com/css/css3_flexbox.asp

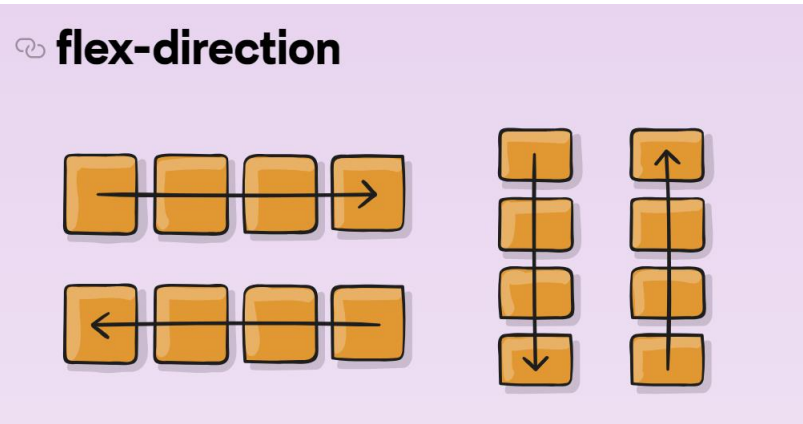
Day 1 – Concetti di Base su CSS

Flexbox – display e direction

Per prima cosa bisogna definire il tipo di display che deve avere il contenitore.

```
.container {  
  display: flex; /* or inline-flex */  
}
```

A questo punto bisogna definire la direzione che si vuole assegnare agli elementi del contenitore.



```
flex-direction: row | row-reverse | column | column-reverse;
```

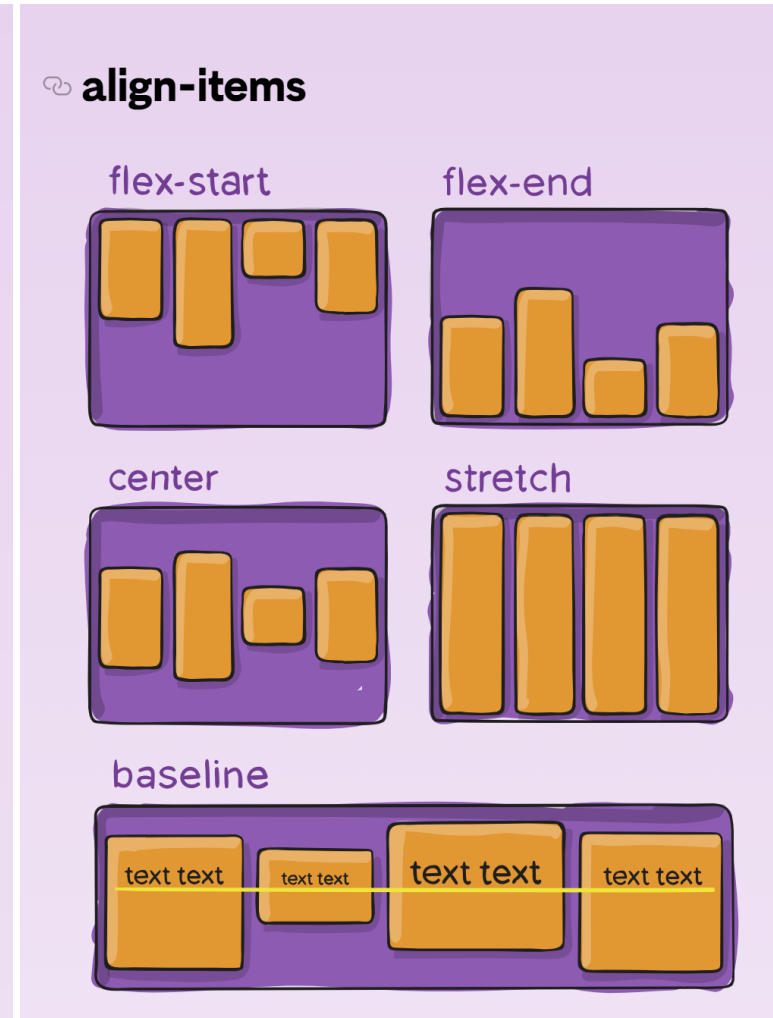
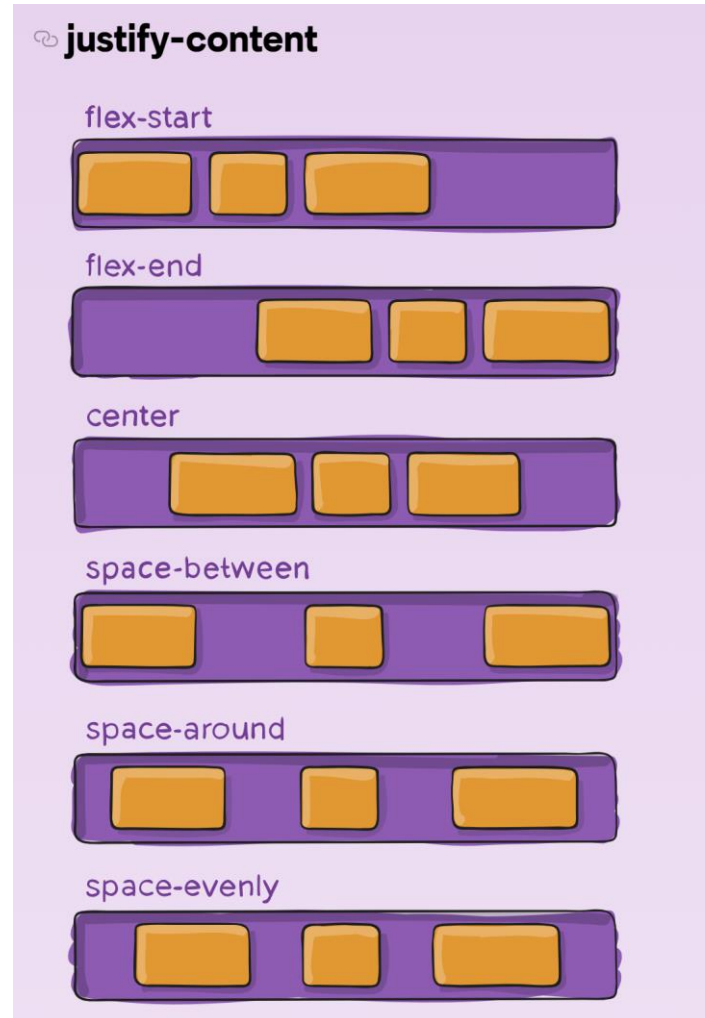
Day 1 – Concetti di Base su CSS

Flexbox - allineamento

Gli elementi del contenitore possono essere allineati sull'asse principale o sull'asse trasversale.

Per allinearli sull'asse principale si utilizza la proprietà **justify-content**.

Per allinearli sull'asse trasversale si utilizza la proprietà **align-items**.



Day 1 – Responsiveness di una pagina Web

Responsiveness di una pagina HTML

- Una pagine web è responsive quando il contenuto si adatta automaticamente a diverse risoluzioni o dispositivi
- *Come si implementa la responsiveness?*
In un contesto progettuale si usano le librerie, perchè prevedono classi già definite e pronte per essere responsive.
Tra le più utilizzate c'è **Bootstrap**



```
<!DOCTYPE html>
<html>
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <style>
    .example {
      padding: 20px;
      color: white;
    }
    /* Extra small devices (phones, 600px and down) */
    @media only screen and (max-width: 600px) {
      .example {
        background: red;
      }
    }
    /* Small devices (portrait tablets and large phones, 600px and up) */
    @media only screen and (min-width: 600px) {
      .example {
        background: green;
      }
    }
    /* Medium devices (landscape tablets, 768px and up) */
    @media only screen and (min-width: 768px) {
      .example {
        background: blue;
      }
    }
    /* Large devices (laptops/desktops, 992px and up) */
    @media only screen and (min-width: 992px) {
      .example {
        background: orange;
      }
    }
    /* Extra large devices (large laptops and desktops, 1200px and up) */
    @media only screen and (min-width: 1200px) {
      .example {
        background: pink;
      }
    }
  </style>
</head>
<body>
  <h2>Typical Breakpoints</h2>
  <p class="example">Resize the window to see how the background color.</p>
</body>
</html>
```

Per approfondimenti: https://www.w3schools.com/css/css_rwd_mediaqueries.asp

HTML + CSS + Responsive

Per sviluppare una UI moderna e responsive ci viene in soccorso un ottimo framework, **Bootstrap**!
Se si usasse lo stile personalizzato, si reinventerebbe la ruota. Utilizzando una libreria si evitano eventuali bug

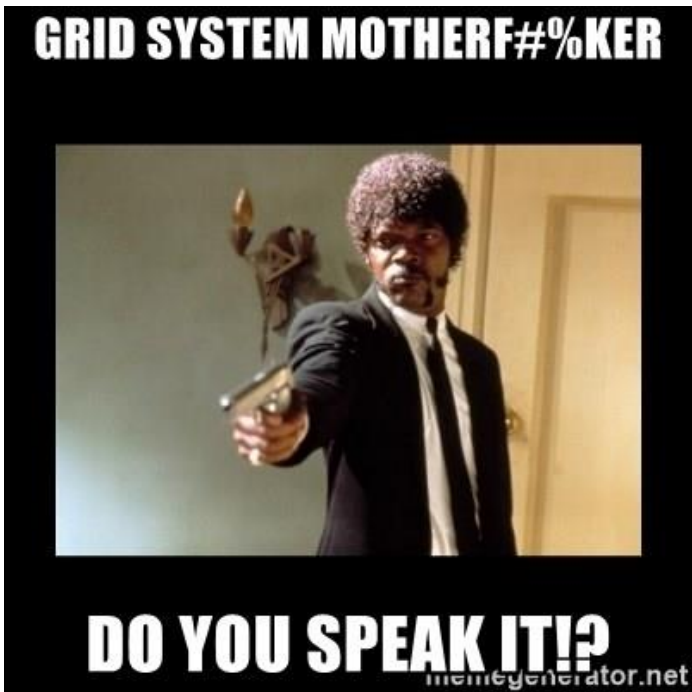


Cos'è Bootstrap e come integrarlo in un progetto?

Bootstrap è un framework che consente di sviluppare pagine web responsive.
Si integra richiamando la libreria nella pagina HTML



Gestire in modo agevole le interfacce responsive



Tutte le componenti di BS sono disponibili nella documentazione ufficiale. Tra gli elementi fondamentali di BS c'è il **grid system** che consente di organizzare gli elementi a schermo in colonne, definendo una larghezza dinamica.

Questo sistema ti permette di dividere la tua pagina in righe e colonne, rendendo più semplice l'organizzazione del contenuto. Ogni riga è **divisa in 12** colonne di uguale larghezza e puoi utilizzare le classi di Bootstrap per definire quante colonne occupa un elemento all'interno della riga. Questo ti permette di creare layout adattivi che si adattano automaticamente a diverse larghezze di schermo.

span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1
span 4				span 4				span 4			
span 4				span 8							
span 6						span 6					
span 12											

Come utilizzare il grid system in pratica?

Column

Column

Column

```
<div class="container">
  <div class="row">
    <div class="col">
      Column
    </div>
    <div class="col">
      Column
    </div>
    <div class="col">
      Column
    </div>
  </div>
</div>
```

Copy

Caso d'uso:

Dividere una pagina web in 3 colonne.

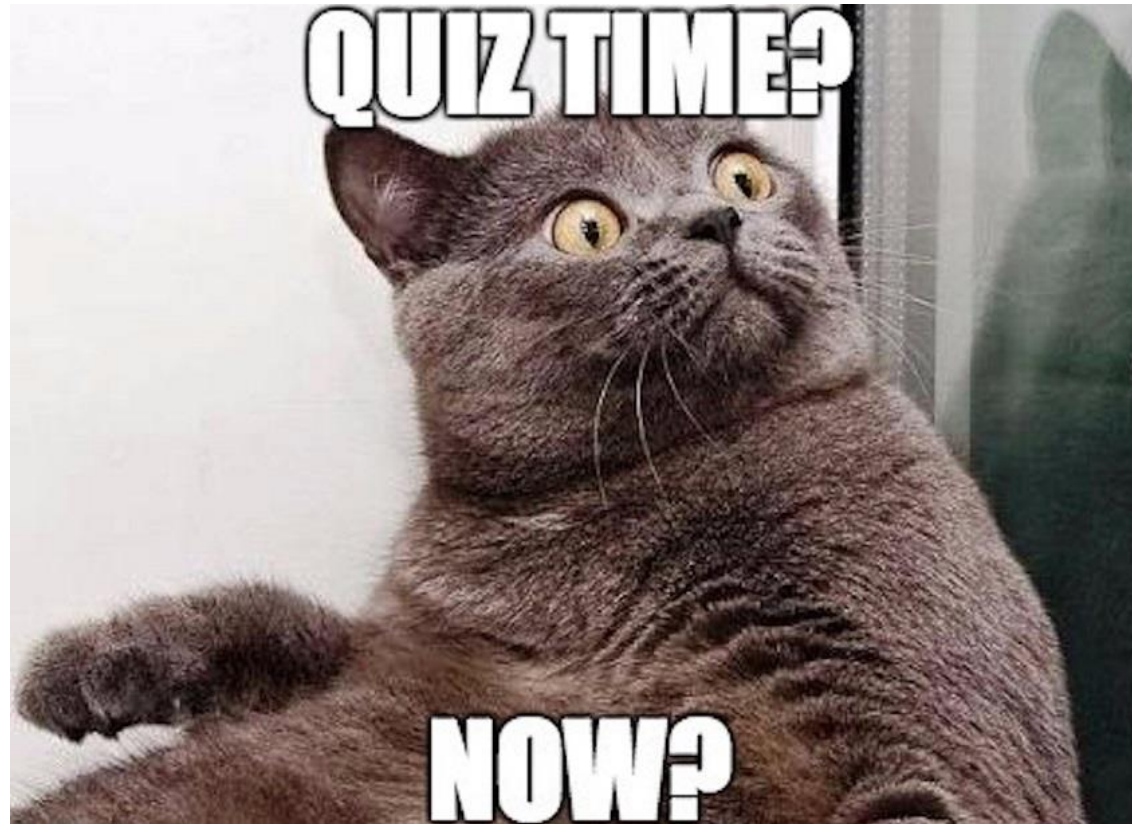
Soluzione:

1. definire un div container **primario**
2. definire un div **riga** con la classe *row*
3. definire tre div **colonne** con la classe *col*

In questo modo la riga verrà divisa in 3 colonne di grandezza identica, quindi $12/3=4$

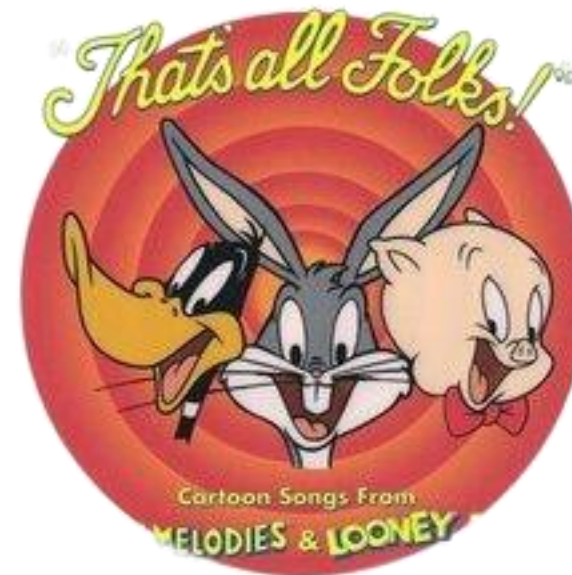
Occhio alle tante componenti utili presenti in bootstrap, come ad esempio i bottoni, selettori, text-box e soprattutto come rendere le *immagini responsive*!

Day 1 – Quiz





Experience.
Create.
Inspire.



Thank You

Vitale Esca

IBM Client Innovation Center
Italy