

CSS – Base

Settimana 1 – Parte 2

Cosa faremo

- Selettori per classi e id
- Alcune proprietà CSS per i font
- Proprietà “display”
- Posizionamento degli elementi nella pagina

Perché


- Essere più precisi quando si selezionano gli elementi HTML
- Personalizzare al meglio le pagine web
- Controllare quanto spazio occupano gli elementi.
- Lavorare con la visibilità degli elementi.
- Spostare gli elementi dalla loro posizione predefinita.

Alla fine della giornata:

- Potrai selezionare qualsiasi elemento nella pagina in modo semplice e preciso
- Sarai in grado di personalizzare il testo della tua pagina
- Imparerai come includere immagini nel tuo documento

Recap veloce

- HTML Tags (div, span, p, img, ul, li, h1 – h6 e altro...)
- Selettori CSS corrispondenti:



```
<div> Div Element </div>
<span> Span Element </span>
<img />
<h1> Header Element </h1>
```



```
div {
  color: red
}
span {
  background-color: blue
}
img {
  width: 100px;
  height: 100px;
}
h1 {
  font-size: 22pt
}
```

Recap

Selettori

Classi & ID

- Quando si lavora con HTML e CSS, la selezione per TAG non è sempre la soluzione migliore.
- Spesso vuoi selezionare solo un elemento
- O solo una certa quantità di elementi
- È qui che CLASSI e ID tornano utili.

Classi & ID

Attenzione alle
differenze!

CLASSI

- Possono ripetersi
- Possono essere multiple
- Selettore: .

ID

- DEVONO essere unici
- Selettore: #

Classi & ID

HTML

```
<div class='sidebar__wrap'>
  <h1 id='title1'> Website Title </h1>
</div>
```

CSS

```
.sidebar__wrap {
  width: 100%;
  height: 70px;
}

#title1 {
  color: purple;
}

/* OR */

div.sidebar__wrap {
  width: 100%;
  height: 70px;
}

h1#title1 {
  color: purple;
}
```

Demo a seguire

Demo

Rendiamolo bellissimo:
A nessuno piace un sito web in Times New Roman e senza immagini!

Font

- L'aspetto generale del testo
 - Ci sono MOLTE proprietà CSS che cambiano il testo e interagiscono con il font
 - Vedremo solo le PRINCIPALI
- Proprietà del carattere:
 - font-family
 - font-size
 - font-weight

Proprietà: font-size

- La dimensione del testo
- Solitamente si misurano in pt (come Office, Google Docs etc.)

Proprietà: font-size

```

<style>
  #big {
    font-size: 35pt
  }
</style>

<h1> New on Amazon Prime Music </h1>
<h1 id='big'> New on Amazon Prime Music </h1>

```

New on Amazon Prime Music



24pt

New on Amazon Prime Music



35pt

Demo

Proprietà: font-family

- La “forma” del testo
- È il "nome del carattere", quindi come Arial, Cambria, Roboto ecc.
- Nativamente accetta solo "font web" (supportati dal browser)
 - E' possibile usare altre font-family tramite Google Fonts (a seguire)
- Trovi la lista dei font web qui: https://www.w3schools.com/cssref/css_websafe_fonts.asp

font-family

```
<style>
  #arial {
    font-family: Arial, Helvetica, sans-serif; /* suggestion from vs code */
  }
  #cambria {
    font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;
  }
  #garamond {
    font-family: Garamond; /* web font */
  }
</style>

<h1>Times New Roman</h1>
<h1 id='arial'>Arial</h1>
<h1 id='cambria'>Cambria</h1>
<h1 id='garamond'>Garamond</h1>
```

Times New Roman

Arial

Cambria

Garamond

Demo

Google Fonts

- Come per immagini, CSS e script, è possibile importare font da fonti esterne
- Puoi trovare Google Fonts qui:
<https://fonts.google.com/>

Use on the web

To embed a font, copy the code into the `<head>` of your html

☒ `<link>` ☐ `@import`

```
<link rel="preconnect" href="https://
fonts.googleapis.com">
<link rel="preconnect" href="https://
fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.
com/css2?family=Roboto:wght@100&displ
ay=swap" rel="stylesheet">
```

CSS rules to specify families

```
font-family: 'Roboto', sans-serif;
```

Demo

Proprietà: font-weight

- Quanto è "pesante" il carattere (== grassetto)
- Accetta sia valori semantici (lighter, regular, bold) che numeri (100 - 900)
- Quando lavori con i numeri, assicurati che il carattere lo supporti, altrimenti ricadrà su quello più vicino

font-weight

```
<style>
/* Be careful when you use numbers instead of semantic values! Not all fonts
support all sizes, and it will "default" to the closest one */
.lighter {
  font-family: Arial, Helvetica, sans-serif;
  font-weight: lighter;
}
.regular {
  font-family: Arial, Helvetica, sans-serif;
  /* font weight defaults to "regular" or 400 */
}
.bold {
  font-family: Arial, Helvetica, sans-serif;
  font-weight: bold;
}
.boldest {
  font-family: Arial, Helvetica, sans-serif;
  font-weight: 900; /* font-weight also accepts numbers, 900 being the highest */
}
</style>

<h1 class='lighter'>Arial</h1>
<h1 class='regular'>Arial</h1>
<h1 class='bold'>Arial</h1>
<h1 class='boldest'>Arial</h1>
```

Arial

- lighter

Arial

- regular

Arial

- bold

Arial

- 900

Demo

Immagini

- Può essere aggiunto con ``
- E' un tag self-closing (NON FARE ` `)
- Le immagini possono provenire da Internet o da una cartella locale



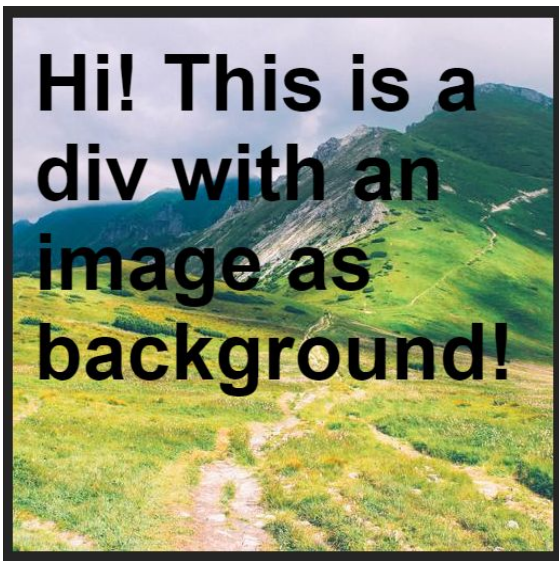
```
  

```

Demo

Immagini come background

Imposta un'immagine come sfondo di un altro elemento (div, p ecc..)



```

<style>
  .image__bg {
    background-image: url(https://picsum.photos/500);
    width: 500px;
    height: 500px;

    font-family: Arial, Helvetica, sans-serif;
    font-weight: bold;
    color: black;
    font-size: 55pt;
  }
</style>

<div class="image__bg">
  Hi! This is a div with an image as background!
</div>
```

Demo

Display

Display

- La proprietà display viene utilizzata per definire il comportamento di rendering degli elementi
 - Ci sono 25 🤖 valori della proprietà display
 - Andremo a vedere solo i PRINCIPALI
 - **Flex avrà una lezione dedicata.**
- Principali tipi di display:
 - Block
 - Inline
 - Inline-Block
 - None
 - **Flex**

Display: block

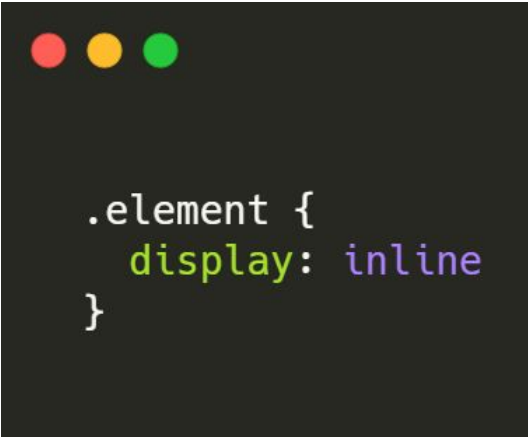
- Rende l'elemento un "elemento di blocco"
- Questo significa che occupa il 100% della pagina in larghezza
- E' il valore di default nei tag **p**, **div**, **h1-h6**, **form**, **ul**, **ol**, **li** (e tanti altri)

```
.element {  
  display: block  
}
```

Demo

Display: inline

- La larghezza dell'elemento è impostata su "fit-content", cioè prende solo lo spazio che gli serve
- Non interrompe il flow della pagina
- Le proprietà width e height non hanno effetto
- Margini e padding applicati al blocco verranno ignorati dagli elementi circostanti!
- Default in span, a



```
.element {  
  display: inline  
}
```

Demo

Display: none

- Nasconde un elemento
- Fa in modo che non occupi spazio nella pagina
- Al contrario di **visibility: hidden**; che mantiene lo spazio per l'elemento



```
.element {  
  display: none  
}
```

Demo

BONUS: Display: inline-block

- La larghezza dell'elemento è impostata su "fit-content"
- Non interrompe il flow della pagina
- Contrariamente a display: inline, accetta le proprietà di larghezza e altezza e margini e padding sono rispettati.

```
.element {  
  display: inline-block  
}
```

Demo

Position

Position

- La **posizione** viene utilizzata per definire COME viene posizionato l'elemento
 - ...ma non DOVE si posiziona
 - Usa le proprietà di top, bottom, left e right per spostare l'elemento
- Tutti i tipi di positioning:
 - Relative
 - Absolute
 - Sticky
 - Fixed
 - Static

Position: static

- Impostazione predefinita. Tutti gli elementi di default hanno position:static.
- Non risente delle proprietà left, right, top, bottom

Position: fixed

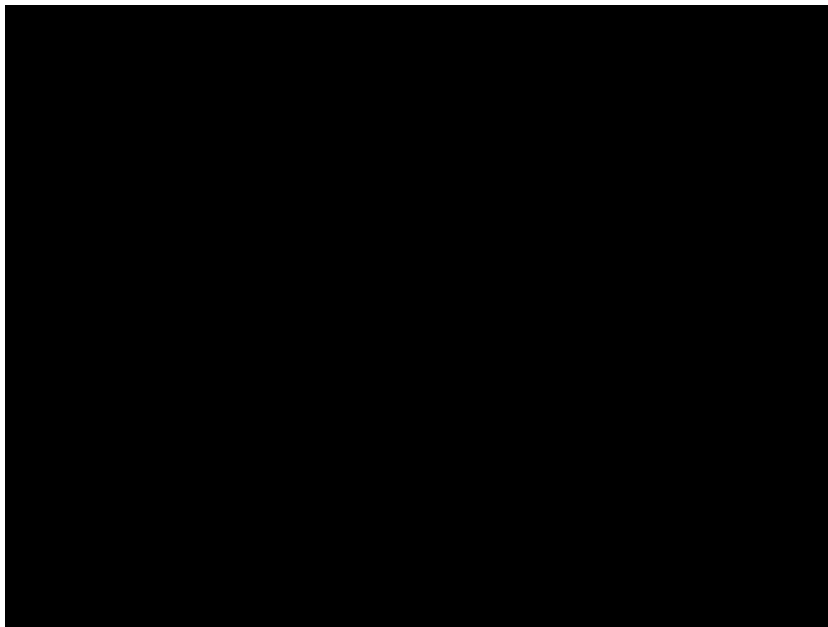
- L'oggetto target viene posizionato in relazione al **viewport (= la finestra del browser)**
- L'elemento rimarrà nello stesso posto se scorriamo verso il basso
- Comportamento di navbar, sidebar, ecc...

```
<style>
h1 {
  position: fixed;
  top: 0;
  left: 0;
}
</style>

<h1> This is staying on the top left corner forever </h1>
```

Position: Fixed - Esempio

Guarda la barra laterale e l'icona chat nell'angolo!



Demo

Position: Relative

- Consente di spostare un elemento rispetto alla sua posizione predefinita.
- Simile a "fixed", ma non rimarrà in posizione una volta avviato lo scorrimento
- Diventa molto utile se usato in coppia con **position: absolute**

```
<style>
  h1 {
    position: relative;
    top: 0;
  }
</style>

<h1> This element will be at the top of the page, but not stay there once we scroll. </h1>
```

Position: Absolute

- L'elemento viene posizionato in base all'antenato "**non statico**" più vicino
- Se non viene trovato alcun antenato non statico, si sposterà in base al **body**
- L'elemento ignora la posizione degli altri elementi e può sovrapparli.
- Diventa molto utile se usato in coppia con **position: relative**

Esempio: Come centrare un div



```
<style>
  .container {
    position: relative;
    border: 2px solid red;
    width: 500px;
    height: 300px;
  }
  .child {
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    border: 2px solid green;
  }
</style>

<div class="container">
  position: relative
  <div class="child">
    position: absolute
  </div>
</div>
```

Demo

Position: Sticky

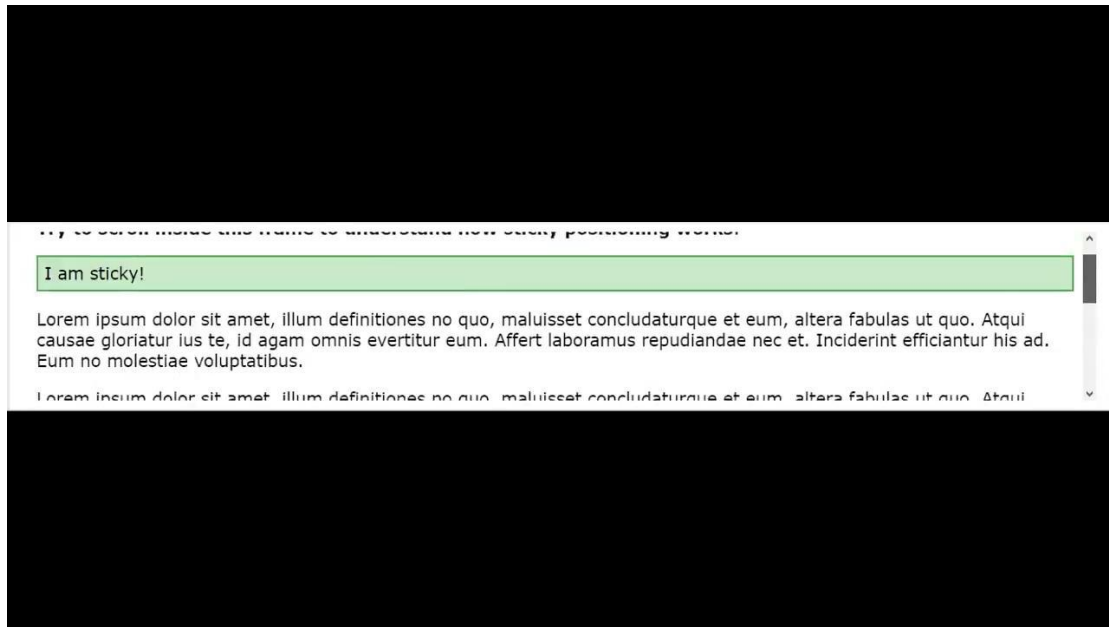
- L'oggetto target viene posizionato in base alla posizione corrente di scroll
- La posizione alterna tra "relative" e "fixed"

```
<style>
  h1 {
    position: sticky;
    top: 0;
  }
</style>

<h1> This element will become "fixed" once it reaches top: 0 </h1>
```

Position: Sticky - Esempio

From: https://www.w3schools.com/css/css_positioning.asp



Q&A

Non essere **timido**, non giudichiamo nessuno



GRAZIE
EPICODE