



boolean



# Simple Web Apps in HTML, CSS e JS





## Web Apps - AI Enhanced

Partiamo con le nozioni base di HTML e CSS che ci serviranno per la struttura e lo stile delle app.

### Cosa impariamo:

- dev tools essenziali
- tag HTML base
- proprietà CSS base
- struttura di una pagina HTML





# Cosa ti serve per programmare



(ma programmare cosa?)



## Browser

**Tra te e l'utente, c'è solo lui: il browser**

Il browser è un programma scritto per comprendere HTML, CSS e JavaScript. Legge (e a volte interpreta!) il codice che scrivono i programmatori e lo disegna a schermo affinchè l'utente possa vederne il risultato.

Negli anni, sono nati diversi browser (*con diversi gradi di aderenza agli standard, RIP Internet Explorer*). Oggi: Chrome, Firefox, Safari e Edge.

La nostra scelta è Chrome, è gratis e puoi scaricarlo da qui:

**<https://www.google.com/chrome/>**





Cosa ti serve per programmare

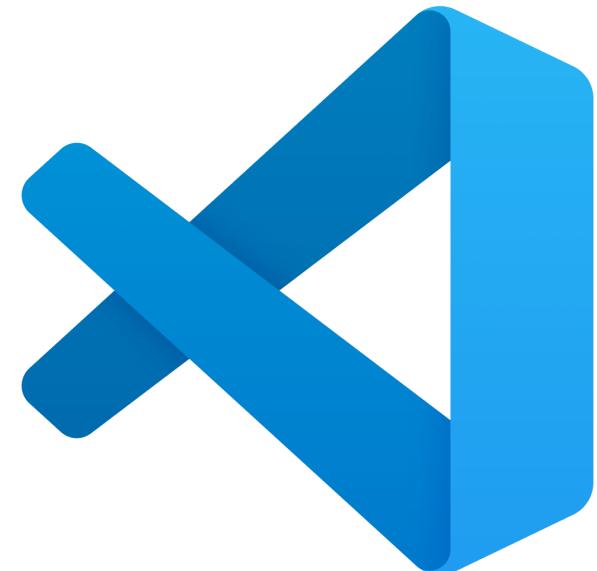
## Editor di Codice (*Code Editor*)

Aiutati a scrivere codice con uno strumento dedicato

Benchè il codice sia "semplicemente testo", è fondamentale utilizzare uno strumento con funzionalità pensate apposta per gli sviluppatori.

Ci sono molti *Code Editor*, tutti simili e tutti validi: la nostra scelta è Visual Studio Code o **VSCode**. E' il più utilizzato, è multipiattaforma ed è gratuito.

Puoi scaricarlo da qui: <https://code.visualstudio.com/>





# HTML

## intro HTML



# Che cos'è l'HTML?

**HyperText Markup Language**

Si occupa della parte di struttura dei contenuti del sito.

Non è un linguaggio di programmazione ma un linguaggio  
di **markup**.

Non essendo un linguaggi di programmazione, non  
contiene costrutti di "logica", come *se...allora (if...else)*

**HTML**





# Un Linguaggio di Markup

## Un esempio di un documento

Elementi ricorrenti:

1 Titolo principale

2 Immagine

3 Sottotitolo

4 Lista puntata

5 Lista numerata

6 Paragrafi e grassetto

7 Separatore

1 Ricetta pancake



3 Ingredienti

- 250 ml di latte
- 200 g di farina
- 1 uovo
- 100 g di zucchero
- 65g di lievito

4

Procedimento

5

1. Mettete in una ciotola l'uovo intero (o due piccole) insieme allo zucchero e sbatteteli con una forchetta oppure con una frusta a mano.
2. Aggiungete il latte a temperatura ambiente e continuate a sbattere solo con la forchetta o con una frusta a mano l'impasto per i pancake senza burro.
3. Aggiungete a poco a poco tutta la farina setacciata e il lievito in polvere per dolci e continuate a mescolare con una frusta a mano in modo che non si formino grumi fino a che otterrete una pastella liscia e omogenea.

Varianti

6

Non serve mettere burro nella padella,  
in questo modo saranno davvero pancake senza burro.

Non schiacciate i pancake veloci dopo averli girati ma lasciateli belli gonfi.

La mia ricetta per pancake Non prevede burro la ricetta quindi non usatelo, non serve.

Ricetta presa da Giallo Zafferano

7



# HTML

## I tag HTML: alcuni esempi

```
1 <h1>Intestazione</h1>
2
3 <p>Inserisci un lungo pezzo di testo qui</p>
4
5 <br>
6
7 <strong>Grassetto</strong>
8
9 <button>Bottone</button>
10
11 <hr>
12
13 <ul>
14   <li>Elemento lista</li>
15   <li>Elemento lista</li>
16 </ul>
17
18
19 <!-- Questo è un commento -->
```



## Intestazione

Inserisci un lungo pezzo di testo qui

### Grassetto

Bottone

- 
- Elemento Lista
  - Elemento Lista



**CSS**

**cascading style sheet**



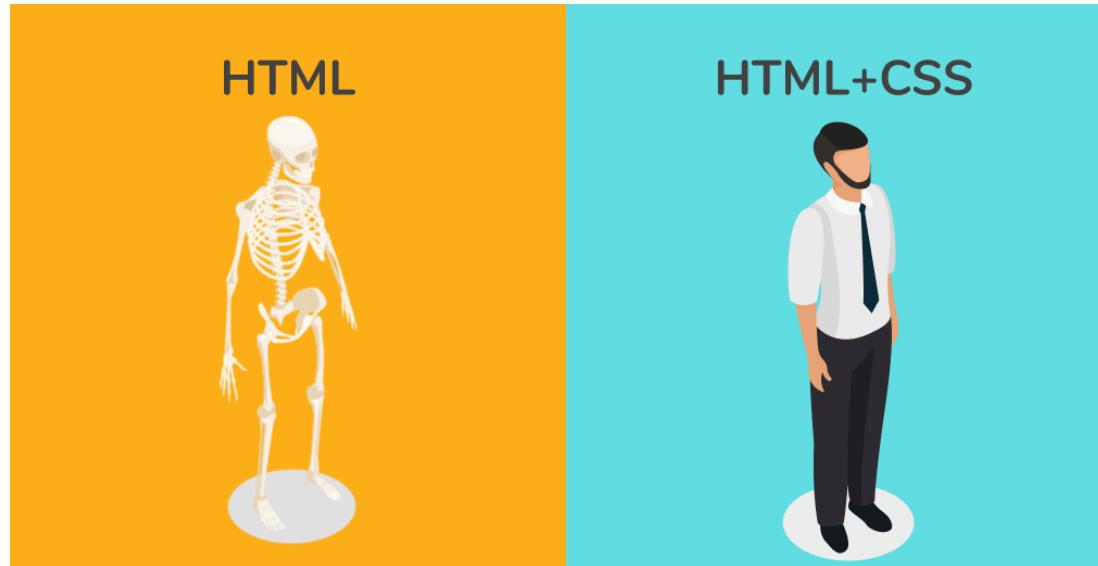
css

## A cosa serve il CSS?

HTML permette di creare la **struttura** della pagina,

I CSS permettono di definire l'**aspetto estetico** dei tag, andando oltre il loro aspetto standard.

Usando i CSS avremo il totale controllo di come appaiano le pagine web, partendo dal colore del testo arrivando alla posizione sullo schermo.



<https://www.interviewbit.com/blog/difference-between-html-and-css/>



CSS

# Le Regole CSS

text-align: center;



## Accesso facile e sicuro ai tuoi contenuti

background-color: blue;



Lavora in modo collaborativo su file e cartelle, archiviali e condividili dal tuo dispositivo mobile o computer

Prova Drive per il lavoro

Vai a Drive

font-weight: bold;



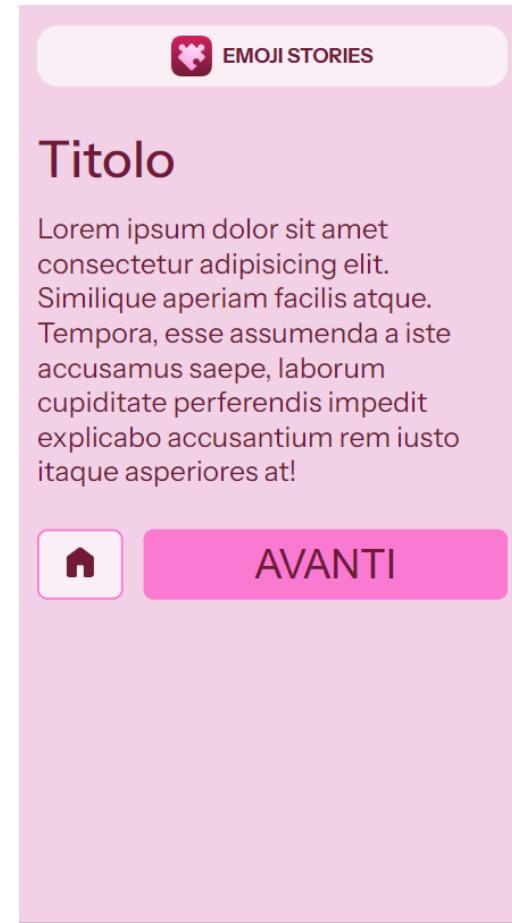
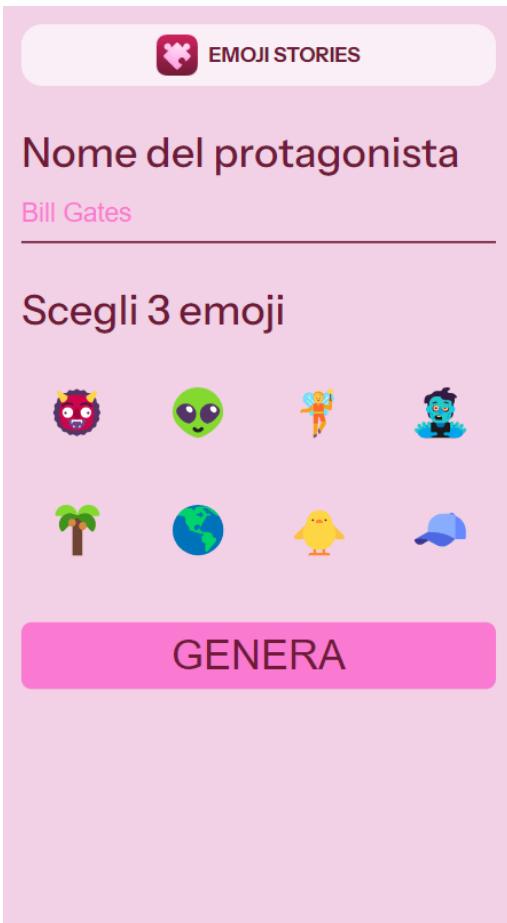
Non hai un account? [Registrati senza alcun costo](#)





# Emoji Stories

Con questi linguaggi costruiremo le 3 schermate che serviranno alla nostra prima app





# CHECKPOINT

Prepariamoci a lavorare: setup di progetto



# HTML

## i tag fondamentali



# La base del Web

## Il linguaggio HTML

I **tag** rappresentano i vari elementi con delle **etichette** di apertura e chiusura

**<tag>** delimita l'inizio della parte di testo  
**</tag>** delimita la fine della parte di testo

A seconda di quello che si scrive al posto di "tag",  
si avrà un effetto e un significato diverso.



```
1 <tag>Testo</tag>
```



# HTML

## Struttura di base

Ogni pagina HTML deve avere questa struttura di base:

- Dichiarazione tipo di documento
- Tag **<html>**
- Tag **<head>** e **<body>**

```
● ● ●  
1  <!DOCTYPE html>  
2  
3  <html>  
4  
5      <head></head>  
6  
7      <body></body>  
8  
9  </html>
```



# HTML

## I tag più utilizzati

```
1 <h1>Intestazione</h1>
2
3 <p>Inserisci un lungo pezzo di testo qui</p>
4
5 <br>
6
7 <strong>Grassetto</strong>
8
9 <button>Bottone</button>
10
11 <hr>
12
13 <ul>
14   <li>Elemento lista</li>
15   <li>Elemento lista</li>
16 </ul>
17
18
19 <!-- Questo è un commento -->
```



## Intestazione

Inserisci un lungo pezzo di testo qui

### Grassetto

Bottone

- 
- Elemento Lista
  - Elemento Lista



# Self Closing Tags

Alcuni tag **non hanno etichetta di chiusura**.

Questo perchè, per la natura degli elementi rappresentati, non hanno necessità di delimitare la loro fine

● ● ●

```
1 <p>Block tag</p>
2
3 <strong>Inline tag</strong>
4
5
6
7
8
9 <!-- Self closing tag -->
10 <br>
```



# Tag "semantici"

Codice come developer professionisti



```
1 <header>
2   <h1>Titolo importante</h1>
3 </header>
4
5 <main>
6   <p>Paragrafo con parola <strong>importante</strong>. </p>
7 </main>
8
9 <footer>Made with Love by Boolean</footer>
```



Titolo importante

Paragrafo con parola **importante**.

Made with Love by Boolean



# HTML - Attributi

Gli attributi sono delle informazioni aggiuntive che si possono inserire nel tag di apertura degli elementi HTML.

Tutti i tag HTML possono avere uno o più attributi, definiti sempre con la sintassi **nome="valore"**.

Possono essere universali (applicabili su qualunque tag) o specifici di alcuni tag.



```
1 <tag attribute="value">testo</tag>
```



# Immagini

## Tag <img> con attributo src

Per visualizzare un'immagine all'interno della pagina, utilizziamo il tag <**img**>.

Il suo attributo **src** indica al browser il percorso del file da visualizzare.

Opzionalmente è possibile aggiungere l'attributo **alt** per definire un testo alternativo all'immagine: non è obbligatorio ma è una **buona prassi**.



```
1 
```



# Alcuni aspetti sui tag

## Tag con display BLOCK

Un elemento con display "block" occupa una intera linea; gli elementi successivi e precedenti occuperanno una nuova linea.

## Tag con display INLINE

Gli elementi occupano solamente lo spazio del loro contenuto; Altri elementi inline possono affiancarsi sulla stessa linea.

```
● ● ●  
1 <!-- Alcuni tag con display block -->  
2  
3 <h1>Block Tag</h1>  
4  
5 <p>Block tag</p>  
6  
7  
8  
9 <!-- Alcuni tag con display inline -->  
10  
11 <strong>Inline tag</strong>  
12  
13 <button>Bottone</button>
```



# Tag generici

A volte, non esistono tag per ciò che vorremmo rappresentare, per cui possiamo usare tag generici e poi manipolarne l'aspetto con CSS!

## <div>

E' un tag generico, non rappresenta un elemento specifico, ma i suoi comportamenti sono di tipo **block**.

Viene spesso usato per raggruppare più tag insieme, con diversi vantaggi soprattutto in combinazione col CSS

## <span>

E' un tag generico, non rappresenta un elemento specifico, ma i suoi comportamenti sono di tipo **inline**.

Viene spesso usato per dare stili particolari a determinate porzioni di testo grazie al CSS.

```
1 <!-- tag block generico -->
2 <div></div>
3
4 <!-- tag inline generico -->
5 <span></span>
6
7
8
9 <!-- Il div raggruppa spesso più elementi -->
10 <div>
11   <p>Lorem ipsum dolor sit amet</p>
12   <button>Clicca qui</button>
13 </div>
```



# Attributo class

L'attributo **class** è un attributo universale che viene spesso usato sui tag generici (ma non solo) per dar loro un significato più comprensibile.

Viene anche utilizzato per accomunare più elementi appartenenti alla stessa "categoria" e/o distinguere tag simili in base al loro ruolo in pagina

E' inoltre utilissimo per:

- definire regole di stile comuni con CSS
- identificare più elementi con JS

```
1 <div class="buttons">
2   <button>PLAY</button>
3   <button>PAUSE</button>
4   <button>STOP</button>
5 </div>
6
7
8
9
10 <ul class="students-list">
11   <li class="student">Francesco</li>
12   <li class="student">Gioele</li>
13   <li class="student">Giorgia</li>
14 </ul>
15
16 <ul class="teachers-list">
17   <li class="teacher">Marco</li>
18   <li class="teacher">Luigi</li>
19   <li class="teacher">Clelia</li>
20 </ul>
21
```



# CHECKPOINT



# CSS

istruzioni per l'uso



CSS

# Le Regole CSS

text-align: center;



## Accesso facile e sicuro ai tuoi contenuti

background-color: blue;



Lavora in modo collaborativo su file e cartelle, archiviali e condividili dal tuo dispositivo mobile o computer

Prova Drive per il lavoro

Vai a Drive

font-weight: bold;



Non hai un account? [Registrati senza alcun costo](#)





CSS

# Sintassi CSS

```
● ● ●  
1 selettore {  
2     proprieta: valore;  
3     proprieta: valore;  
4 }  
5  
6  
7 /* Questo è un commento */
```



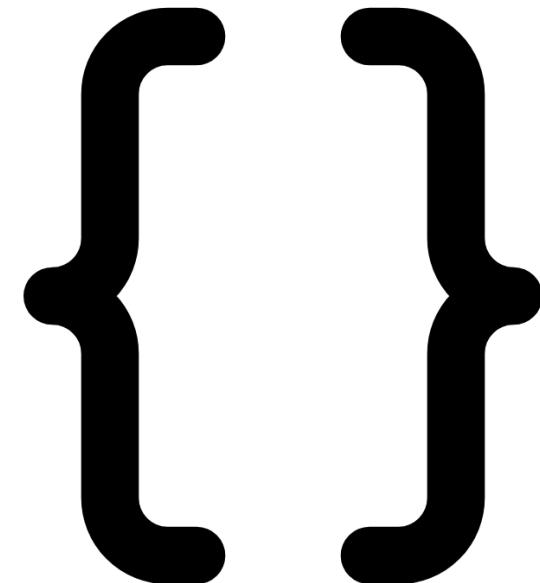
# Dove trovo le parentesi graffe nella tastiera? 🤔

## Su Mac:

- Parentesi graffa aperta: premere Alt Gr + [
- Parentesi graffa chiusa: premere Alt Gr + ]

## Su Windows:

- Parentesi graffa aperta: premere Alt Gr + Shift + [
- Parentesi graffa chiusa: premere Alt Gr + Shift + ]



## Selezione con \*

Il selettori universale serve a dare delle regole generali a tutti gli elementi.

Queste regole potranno poi essere sovrascritte da altri selettori inseriti successivamente.

HTML

```
1 <h1>
2     Questo testo sarà rosso
3 </h1>
4
5 <p>
6     e anche questo
7 </p>
```

CSS

```
1 /* I testi di tutti gli elementi saranno rossi */
2
3 *
4     color: red;
5 }
```



CSS

## Selezione con il tag

Tutti i tag di quel tipo avranno le proprietà definite nel css.

In questo caso, tutti i paragrafi saranno di colore rosso.

HTML

```
1 <h1>Questo titolo non sarà rosso</h1>
2
3 <p>Questo testo sarà rosso e così ogni tag p</p>
4
5 <p>Anche questo testo sarà rosso</p>
6
```

CSS

```
1 /* tutti i tag p saranno rossi */
2 p {
3     color: red;
4 }
```



## Selezione con classe

Tutti i tag a cui è stata assegnata la stessa classe avranno le proprietà definite nel css.

In questo esempio, tutti gli elementi con classe "pippo" saranno di colore rosso.

index.html

```
1 <h1 class="pippo">
2     Questo testo sarà rosso e così ogni tag con
3         classe pippo
4 </h1>
5
6 <p class="pippo">
7     Anche questo paragrafo sarà rosso
8 </p>
9
10 <p>
11     questo paragrafo, invece, NON sarà rosso
12 </p>
```

style.css

```
1 /* tutti i tag con classe pippo saranno rossi */
2 .pippo {
3     color: red;
4 }
```



# Dove si scrive il codice CSS?

Separato dall'HTML, ma pur sempre collegato!

In un file dedicato con estensione **.css**, il quale viene collegato alla pagina html nel tag head

index.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <!-- ... -->
5     <link rel="stylesheet" href="style.css">
6 </head>
7 <body>
8
9     <h1>Questo titolo sarà blu</h1>
10
11    <p class="bot">Questo testo sarà rosso</p>
12
13 </body>
14 </html>
```

style.css

```
1 h1 {
2     color: blue;
3 }
4
5 .bot {
6     color: red;
7 }
```



# CHECKPOINT

Collegiamo il nostro foglio di stile!



# CSS

**Colori, stili e tipografia**



# Color

## Il colore del testo

La proprietà **color** permette di impostare il colore del testo.

Il valore viene indicato utilizzando una delle notazioni valide per i colori.



```
1 p {  
2 /* notazione esadecimale */  
3 color: #A2CC85;  
4  
5 /* notazione rgb */  
6 color: rgb(162, 204, 133);  
7 }  
8  
9 /* Nota: ci sono anche altre proprietà che usano il colore ad esempio border e background-color */
```

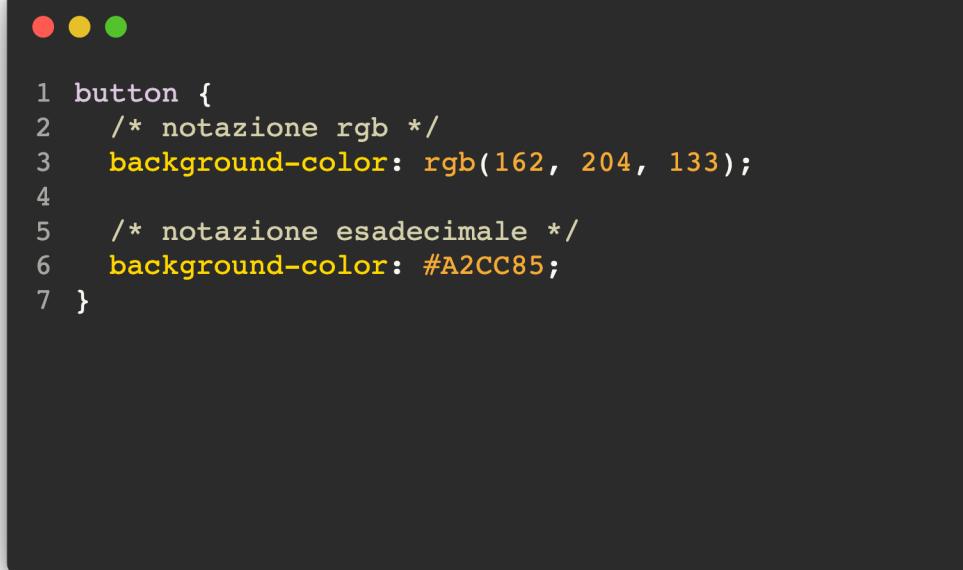


# Background color

## Lo sfondo di un elemento

La proprietà **background-color** permette di impostare il colore di sfondo di un blocco.

Il valore viene indicato utilizzando una delle notazioni disponibili per i colori.



```
1 button {  
2   /* notazione rgb */  
3   background-color: rgb(162, 204, 133);  
4  
5   /* notazione esadecimale */  
6   background-color: #A2CC85;  
7 }
```



# Il Testo

La presentazione del testo può essere modificata in vari modi.

Possiamo ad esempio rendere il testo sottolineato, maiuscolo oppure modificarne l'allineamento.

Allineamento del testo



```
 1 h1 {  
 2   /* grafia del testo */  
 3   text-transform: uppercase;  
 4  
 5   /* allineamento del testo */  
 6   text-align: center;  
 7  
 8 }
```



# Font

## Altre caratteristiche del testo

Il font è lo "stile" delle lettere utilizzate per il testo. Lo stesso termine viene utilizzato anche in editoria.

Con il CSS possiamo andare a modificare l'aspetto del testo in tanti modi, impostando ad esempio, la tipologia di font, la dimensione o lo spessore.

```
● ● ●  
1 p {  
2  
3   /* dimensione del font */  
4   font-size: 12px;  
5  
6   /* nome o tipologia del font */  
7   font-family: Arial, sans-serif;  
8  
9   /* spessore del font */  
10  font-weight: 600;  
11  
12 }
```



# Font Family

## Web Safe Fonts

Font che sono (molto probabilmente) installati sul pc dell'utente e quindi sono "sicuri" da utilizzare (Arial, Verdana, ...) ([web safe fonts](#))

## Google Fonts

Font che non sono installati sul pc dell'utente ma ogni utente dovrà scaricarne una copia da Google Fonts ([fonts.googleapis.com](https://fonts.googleapis.com))

The screenshot shows a grid of font families from Google Fonts. Each row contains a font family name, its designer, and a preview of sample text.

Font Family	Designer	Sample Text
Slabo 27px	John Hudson (1 style)	Silver mist suffused the deck of the ship.
Oswald	Vernon Adams, Kalapi Gajjar, Cyreal (6 styles)	<b>The face of the moon was in shadow.</b>
Source Sans Pro	Paul D. Hunt (12 styles)	She stared through the window at the stars.
Raleway	Multiple Designers (18 styles)	The sky was cloudless and of a deep dark blue.
PT Sans	ParaType (4 styles)	The spectacle before us was indeed sublime.
Roboto Slab	Christian Robertson (4 styles)	Then came the night of the first falling star.

At the bottom left, there is a text input field with placeholder text "Try typing directly into the text fields." and a "GOT IT" button. On the right side, there is a "SEE SPECIMEN" link.



# CSS

## dimensione degli elementi e box model



# Width e Height

Per impostare la larghezza e l'altezza del contenuto dell'elemento

Ci sono numerose unità di misura utilizzabili,  
ma le unità più comunemente usate sono:

- **px** (pixel)
- **%** (percentuale rispetto al tag contenitore)

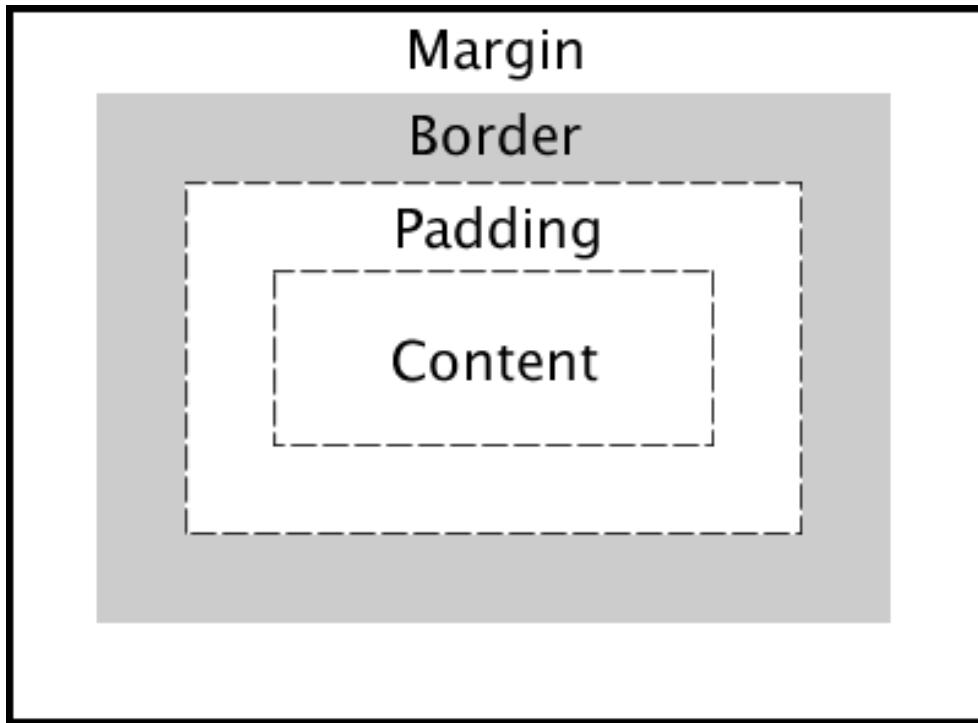


```
1 div {  
2     width: 80%;  
3     height: 80px;  
4 }  
5  
6 /* ESISTONO ANCHE: */  
7  
8     max-width  
9     max-height  
10    min-width  
11    min-height
```



# Bordi, Margini e Padding, cosa sono?

Surprise surprise, non è così banale





# Margin

## Spazio esterno al bordo

Specifica la distanza tra il bordo di un elemento ed il bordo degli elementi adiacenti.

Può essere indicato in uno dei seguenti modi:

- Con un'unità di misura, preferibilmente in pixel
- Come percentuale delle dimensioni del "contenitore"

Se necessario, è possibile definire i margini dei quattro lati singolarmente, per poterli stilare in modo differente e indipendente.

```
1 div {  
2   /* Specifichiamo tutti i lati */  
3   margin: 10px;  
4  
5   /* Specifichiamo sopra/sotto e destra/sinistra */  
6   margin: 10px 5px;  
7  
8   /* Spceifichiamo ogni lato, in senso orario */  
9   margin: 10px 5px 20px 15px;  
10  
11  /* Specifichimo solo un lato */  
12  margin-right: 20px;  
13 }
```



# Padding

## Spazio interno al bordo

Il **padding** imposta la distanza tra il contenuto ed il bordo del blocco.

Può essere indicato in uno dei seguenti modi:

- Con un'unità di misura, preferibilmente in pixel
- Come percentuale delle dimensioni del "contenitore"

Valgono le stesse regole già dette per **margin** per quanto riguarda impostare singolarmente ogni lato e sulle forme abbreviate.

```
1 div {  
2   /* Specifichiamo tutti i lati */  
3   padding: 10px;  
4  
5   /* Specifichiamo sopra/sotto e destra/sinistra */  
6   padding: 10px 5px;  
7  
8   /* Spcecifichiamo ogni lato, in senso orario */  
9   padding: 10px 5px 20px 15px;  
10  
11  /* Specifichimo solo un lato */  
12  padding-right: 20px;  
13 }
```



Di default le dimensioni del padding si sommano alle dimensioni di width e height del blocco, quindi aumentano le dimensioni totali del blocco.



## Box Sizing

### La rivoluzione nel calcolo del dimensionamento dei blocchi

Prima dell'introduzione del concetto di box-sizing, il dimensionamento dei blocchi avveniva solo in una modalità:

width ed height determinano la dimensione del contenuto, e padding e border-width vengono ad esso aggiunti per stabilire lo spazio realmente occupato dal blocco.

Adesso è possibile utilizzare due metodi diversi di **box-sizing**:

- **content-box**  
la "scomoda" modalità di default, in cui **padding e border si sommano** a width e height.
- **border-box**  
**padding e border vengono assorbiti** da width e height.



# CSS Reset

## La prima regola CSS da scrivere

E' buona norma iniziare ogni CSS con una regola che "azzerà" margin, padding e imposta il box-sizing a border-box.

In questo modo, non subiremo le scelte "preconfezionate" dei browser e decideremo noi margin e padding di ogni elemento.

Ovviamente, non è obbligatorio ma è un **ottimo consiglio**

```
* {  
    margin: 0;  
    padding: 0;  
    box-sizing: border-box;  
}  
  
/* Le regole successive potranno  
sovrascriverle quando serve! */
```



# **CSS**

## **flexbox**

# CSS Flexbox

## Che cosa fa Flexbox?

Il CSS3 **FlexBox** è una modalità di layout che prevede la disposizione degli elementi in una pagina in modo «flessibile», con la possibilità di adattare il contenuto più facilmente a schermi e dispositivi diversi.

Gli elementi che vogliamo siano disposti in modo flessibile devono essere contenuti in un **genitore** «Flex», detto «**Flex-Container**».

Gli elementi “**figli**” in un contenitore «Flex» sono detti «**Flex-Item**».

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

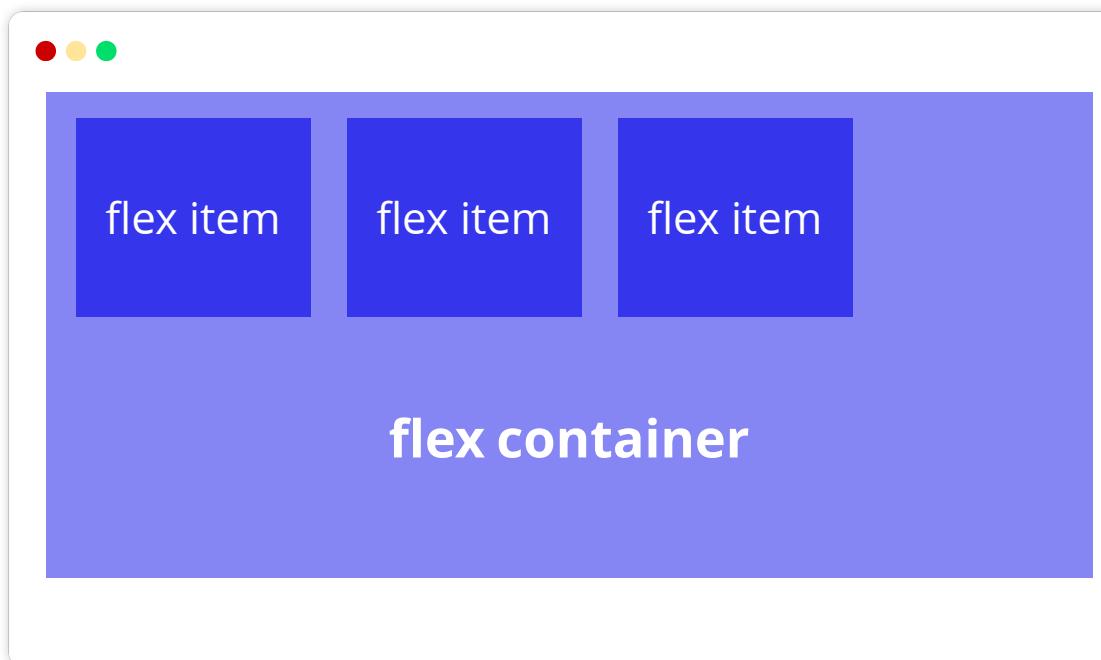
# CSS Flexbox

**Display: flex è una proprietà del genitore!**

La proprietà **display: flex** va data al tag genitore (**flex-container**).

I suoi figli diventeranno automaticamente **flex-item**.

**qualsiasi tipo di elemento verrà sistemato sulla stessa riga.**



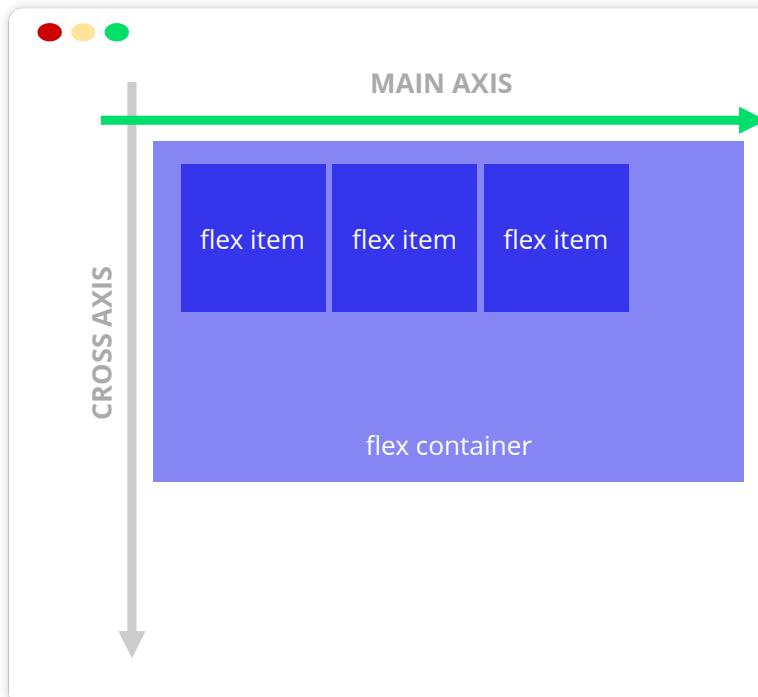
```
1 <div class="container-flex">
2   <div class="item"></div>
3   <div class="item"></div>
4   <div class="item"></div>
5 </div>
```

```
1 .container-flex {
2   display: flex;
3 }
```

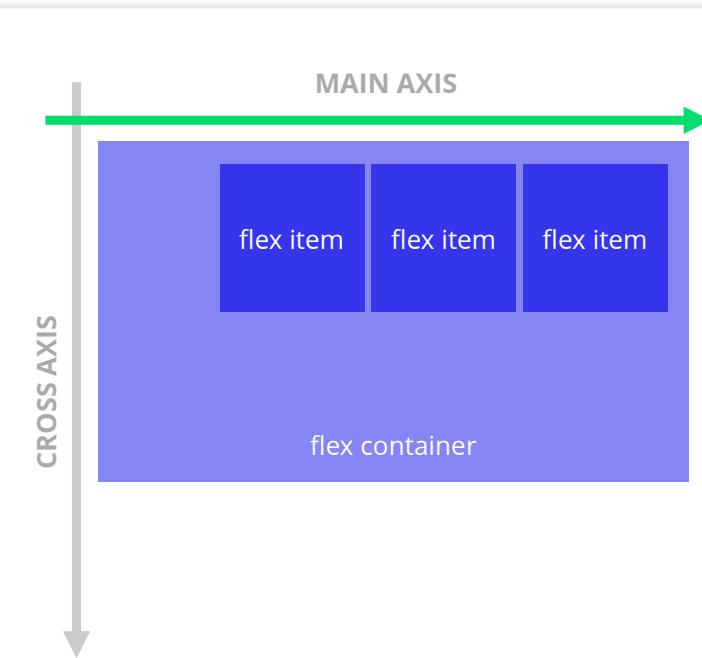
# Possiamo allineare gli item sul Main-Axis

**justify-content** - Flex-container rule

`justify-content: flex-start;`



`justify-content: flex-end;`



**flex-direction: row**

# CSS Flexbox

## justify-content - Flex-container rule

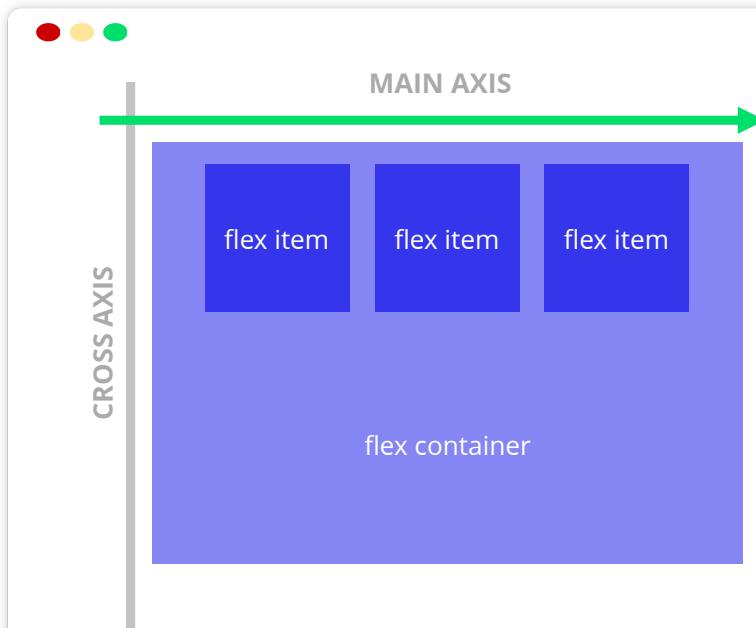
Questa proprietà definisce in che modo i Flex-Item sono disposti lungo il **Main Axis**.  
Può assumere principalmente i seguenti valori:

- flex-start
- flex-end
- center
- space-between
- space-around

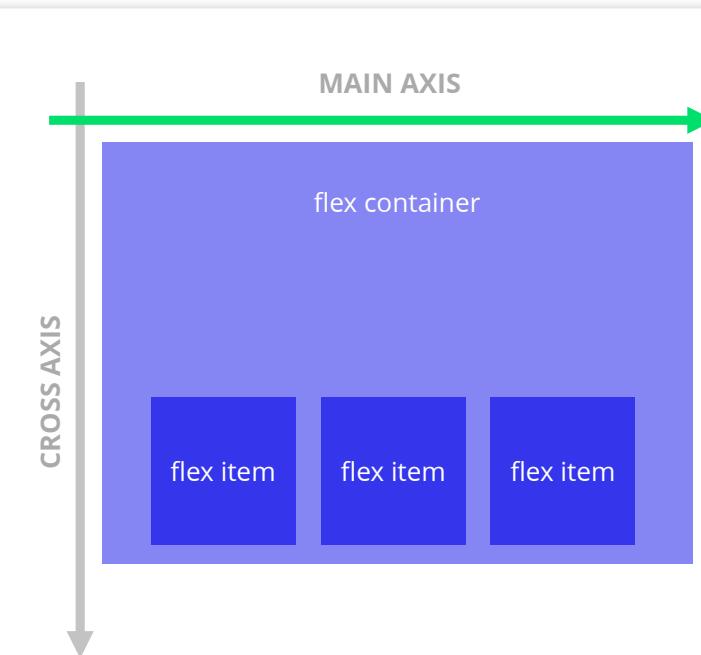
# Possiamo allineare gli item sul Cross-Axis

**align-items** - Flex-container rule

`align-items: flex-start;`



`align-items: flex-end;`



**flex-direction: row**

# CSS Flexbox

## align-items - Flex-container rule

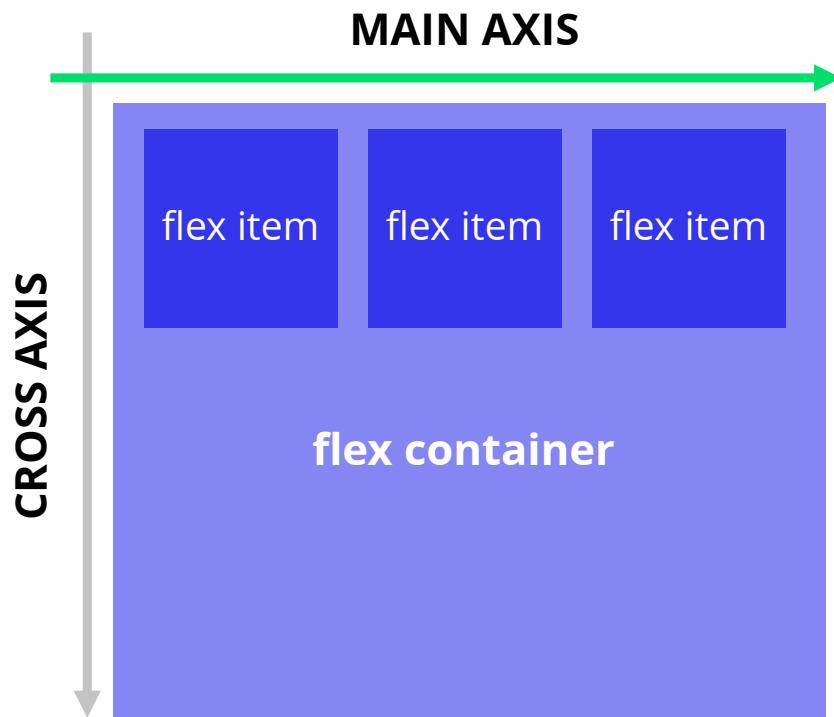
Questa proprietà definisce in che modo i Flex-Item sono disposti lungo il **Cross Axis**.

Può assumere principalmente i seguenti valori:

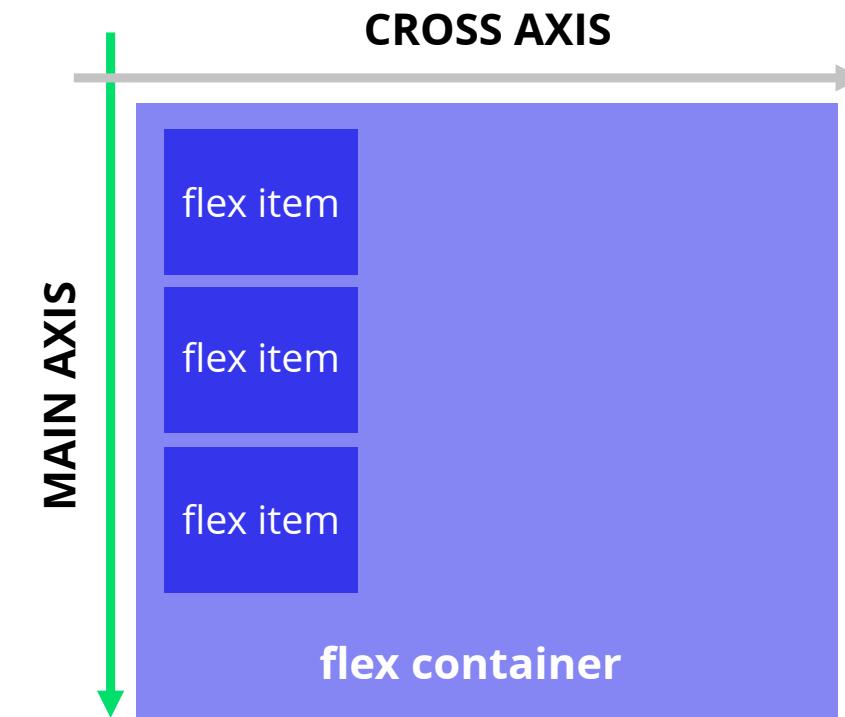
- flex-start
- flex-end
- center
- stretch

# CSS Flexbox

`flex-direction:row`



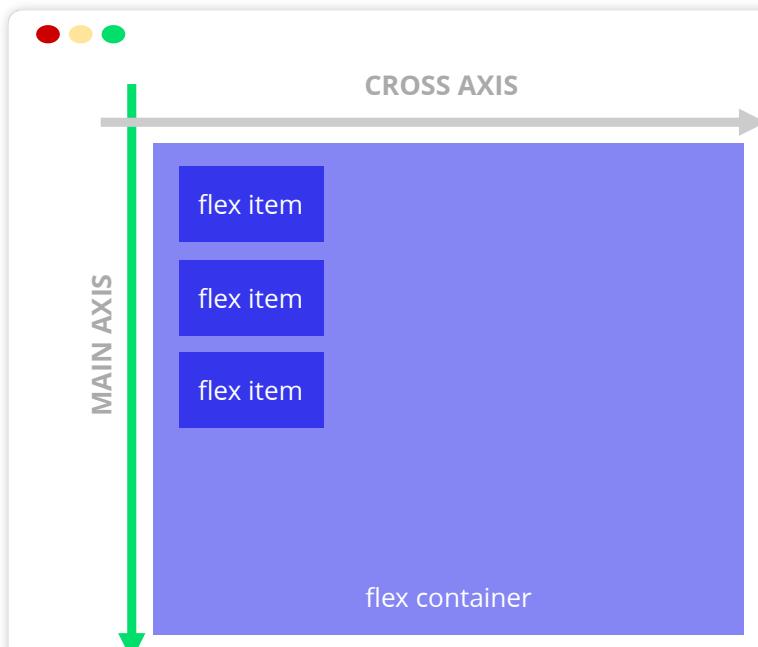
`flex-direction: column`



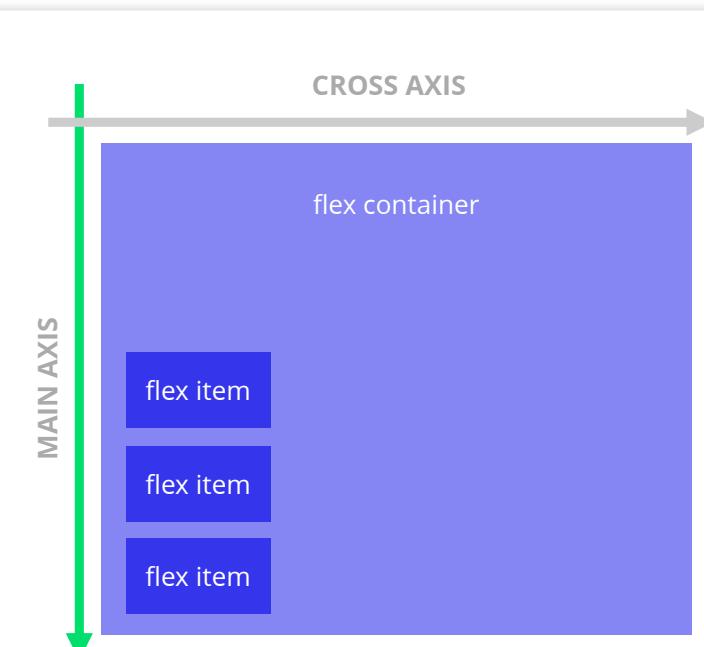
# Se cambiamo la direction tutto si sposta!

**justify-content** - Flex-container rule

`justify-content: flex-start;`



`justify-content: flex-end;`

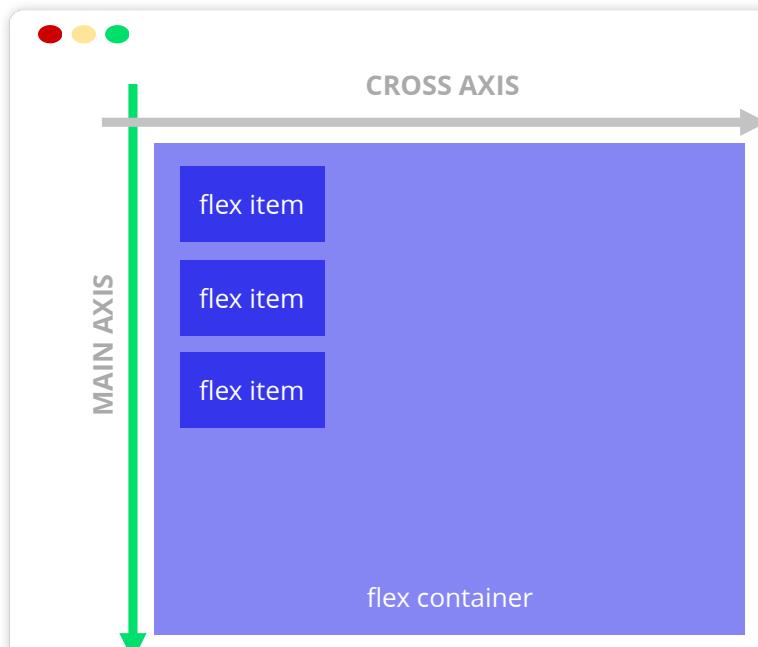


**flex-direction: column**

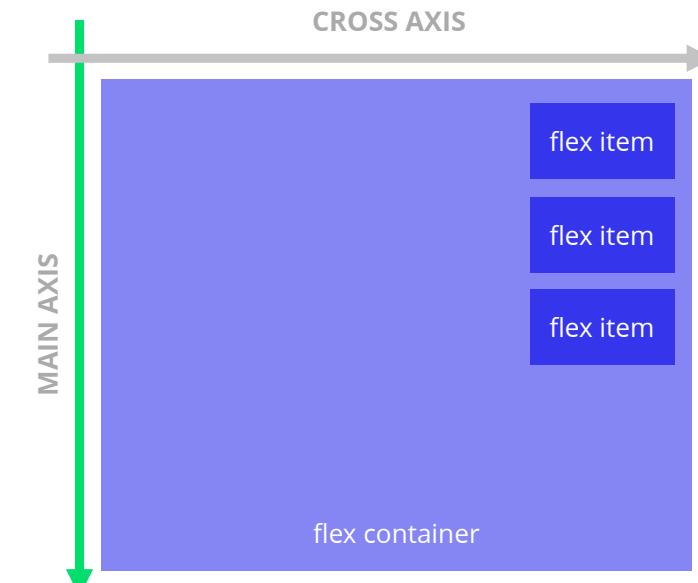
# Se cambiamo la direction tutto si sposta!

**align-items** - Flex-container rule

`align-items: flex-start;`



`align-items: flex-end;`



**flex-direction: column**



CSS

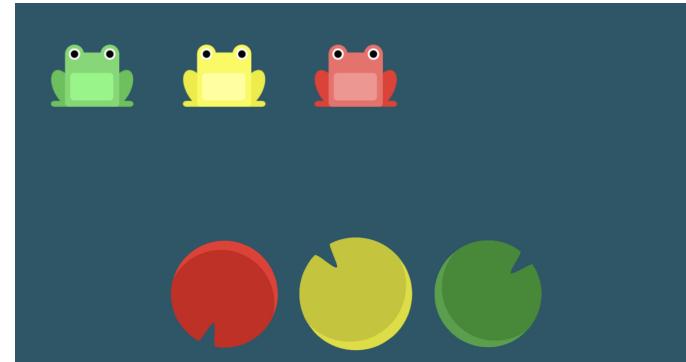
# Vuoi fare pratica con flexbox?

Prova questi mini-giochi!

## Flexfroggy

Aiuta Froggy e i suoi amici scrivendo codice CSS!

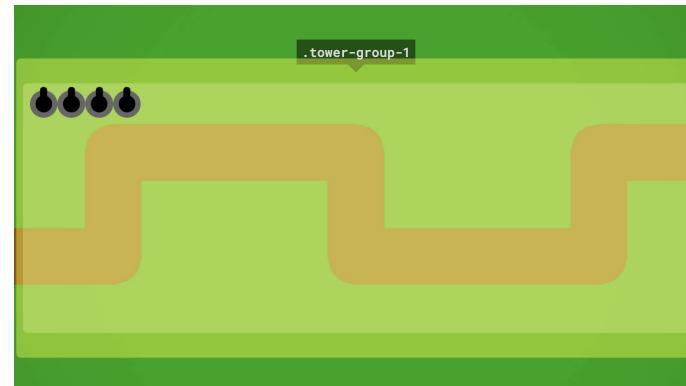
<https://flexboxfroggy.com/#it>



## Flexbox Defence

Tower defence game basato su flexbox.

<http://www.flexboxdefense.com/>





# Dev Hints



# Alcune risorse per sviluppatori

## Personalizza l'interfaccia

Crea un tema personalizzato e cambia sfondo, font, colori o immagini.

### Unsplash

DB di immagini gratuite ad alta risoluzione.

<https://unsplash.com/it>

### Google Fonts

Web Fonts supportati dai browser

<https://fonts.google.com/>

### Colors

Palette generator per la scelta dei colori.

<https://colors.co/>

### SVG Repo

Raccolta di icone gratuite anche in svg.

<https://www.svgrepo.com/>