

**Lo sviluppo delle
competenze di
sviluppatore front-
end**





Hello!

Alessandro Pasqualini

Full-stack developer, appassionato di
programmazione e sviluppo software in generale,
amo le serie tv e i gatti

1

Responsive



Responsive

Da Wikipedia:

"Il **design responsivo**,^[1] o **responsive web design (RWD)**, indica una tecnica di web design per la realizzazione di siti in grado di adattarsi graficamente in modo automatico al dispositivo coi quali vengono visualizzati (computer con diverse risoluzioni, tablet, smartphone, cellulari, web tv), riducendo al minimo la necessità dell'utente di ridimensionare e scorrere i contenuti."



Design responsive vs Design adattivo

Design responsive

Un sito responsivo si adatta esclusivamente alla forma e dimensione dello schermo del dispositivo. La pagina web deve adattarsi in modo fluido alle dimensioni dello schermo.

Es. Facebook e Instagram



Design responsive vs Design adattivo

Design adattivo

Una soluzione adattiva è ottimizzata completamente per i device mobili, ciò garantisce una migliore user experience. Esistono più "versioni" della stessa pagina web: una per i dispositivi mobile, una per i dispositivi desktop, etc. È possibile anche che alcune funzionalità non siano presenti nella versione mobile.

es. Google e Amazon



Design responsive vs Design adattivo

Design adattivo

Una soluzione adattiva è ottimizzata completamente per i device mobili, ciò garantisce una migliore user experience. Esistono più “versioni” della stessa pagina web: una per i dispositivi mobile, una per i dispositivi desktop, etc. È possibile anche che alcune funzionalità non siano presenti nella versione mobile.

es. Google e Amazon



Mobile first vs Desktop first

Mobile first e **desktop first** sono due strategie di sviluppo responsive.

Mobile first

La pagina web viene sviluppata inizialmente per un dispositivo mobile e successivamente "adattata" per gli schermi più grandi.

Desktop first

Esattamente il contrario: la pagina è sviluppata concentrandosi su un dispositivo desktop e poi "adattata" per schermi più piccoli



Mobile first vs Desktop first

Quale scegliere?

Mobile first: se il traffico generato dal sito web proviene principalmente da dispositivi mobile.

Desktop first: se il traffico generato dal sito web proviene in larga parte da dispositivi desktop.

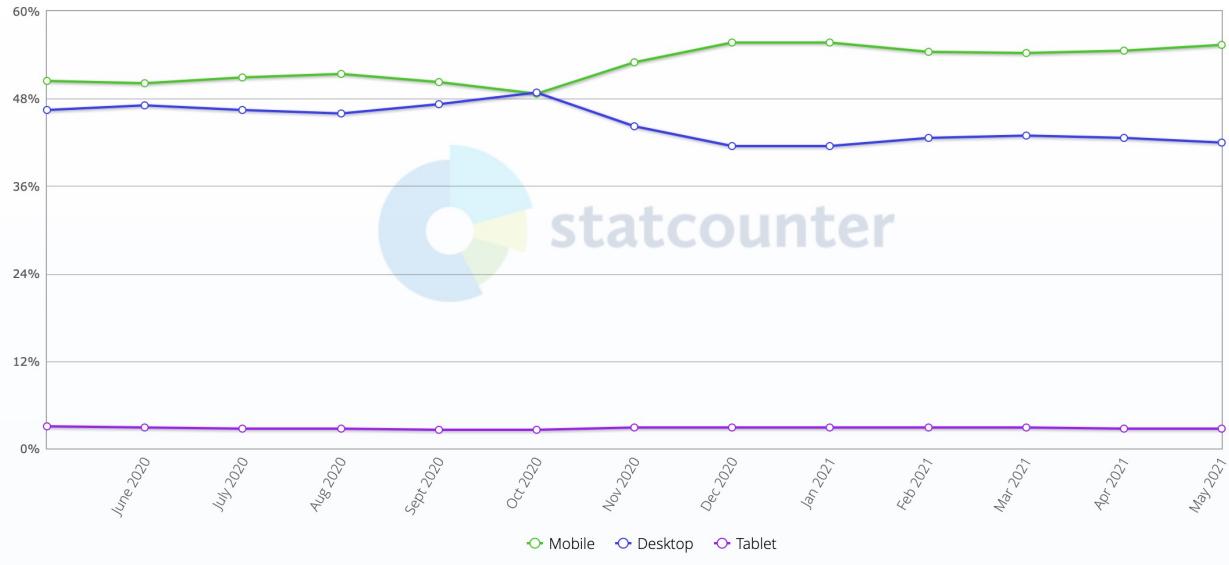
L'idea è che la pagina web deve essere ottimizzata per la maggior parte dei visitatori.



Mobile first vs Desktop first

Desktop vs Mobile vs Tablet Market Share Worldwide
May 2020 - May 2021

Edit Chart Data



2

Responsive in bootstrap



Appriccio mobile first

Bootstrap 4 è sviluppato utilizzando un approccio **mobile first**.

I suoi componenti e le sue classi sono sviluppate e ottimizzate per dispositivi mobile e poi «adattate» per dispositivi desktop.

La maggior parte dei componenti sono sviluppato usando la media query **min-width**

In alcuni casi, per alcuni componenti complessi, viene usata la media query **max-width** (tipica di approcci desktop-first)



Breakpoints

Un **breakpoint** è un punto, su una linea ideale che parte da 0, in cui si verifica una qualche **modifica (tramite i CSS) al layout della pagina**

es.

```
@media (min-width: 768px) {  
    ...  
}
```



Breakpoints di bootstrap

```
// Extra small devices (portrait phones, less than 576px)
// No media query for `xs` since this is the default in Bootstrap

// Small devices (landscape phones, 576px and up)
@media (min-width: 576px) { ... }

// Medium devices (tablets, 768px and up)
@media (min-width: 768px) { ... }

// Large devices (desktops, 992px and up)
@media (min-width: 992px) { ... }

// Extra large devices (large desktops, 1200px and up)
@media (min-width: 1200px) { ... }
```



Es. breakpoints

	Extra small $<576\text{px}$	Small $\geq 576\text{px}$	Medium $\geq 768\text{px}$	Large $\geq 992\text{px}$	Extra large $\geq 1200\text{px}$
<code>.container</code>	100%	540px	720px	960px	1140px
<code>.container-sm</code>	100%	540px	720px	960px	1140px
<code>.container-md</code>	100%	100%	720px	960px	1140px
<code>.container-lg</code>	100%	100%	100%	960px	1140px
<code>.container-xl</code>	100%	100%	100%	100%	1140px
<code>.container-fluid</code>	100%	100%	100%	100%	100%



Display

Bootstrap permette di modificare la proprietà display attraverso delle classi predefinite.

```
.d-{value}  
.d-{breakpoint}-{value}
```

I breakpoint sono: sm, md, lg, e xl

Mentre i valori per **value** sono: none, inline, inline-block, block, table, table-cell, table-row, flex, inline



Display

Le classi

.d-none

.d-{breakpoint}-none

Sono usate per nascondere del contenuto in determinati dispositivi.

Quando usiamo i breakpoint bisogna ricordare che si applicano dal breakpoint scelto fino ad xl. Es. d-none nasconde tutto, d-md-none nasconde il contenuto per md, lg e xl ma lo farà vedere per xs e sm.

3

Utility class di bootstrap

Position

```
<div class="position-static">...</div>
<div class="position-relative">...</div>
<div class="position-absolute">...</div>
<div class="position-fixed">...</div>
<div class="position-sticky">...</div>
```

Con queste classi è possibile modificare la proprietà position degli elementi senza scriverla direttamente nei CSS





Float e clear

Example Button floated left

Example Button floated right

```
<div class="bg-info clearfix">
  <button type="button" class="btn btn-secondary float-left">...</button>
  <button type="button" class="btn btn-secondary float-right">...</button>
</div>
```

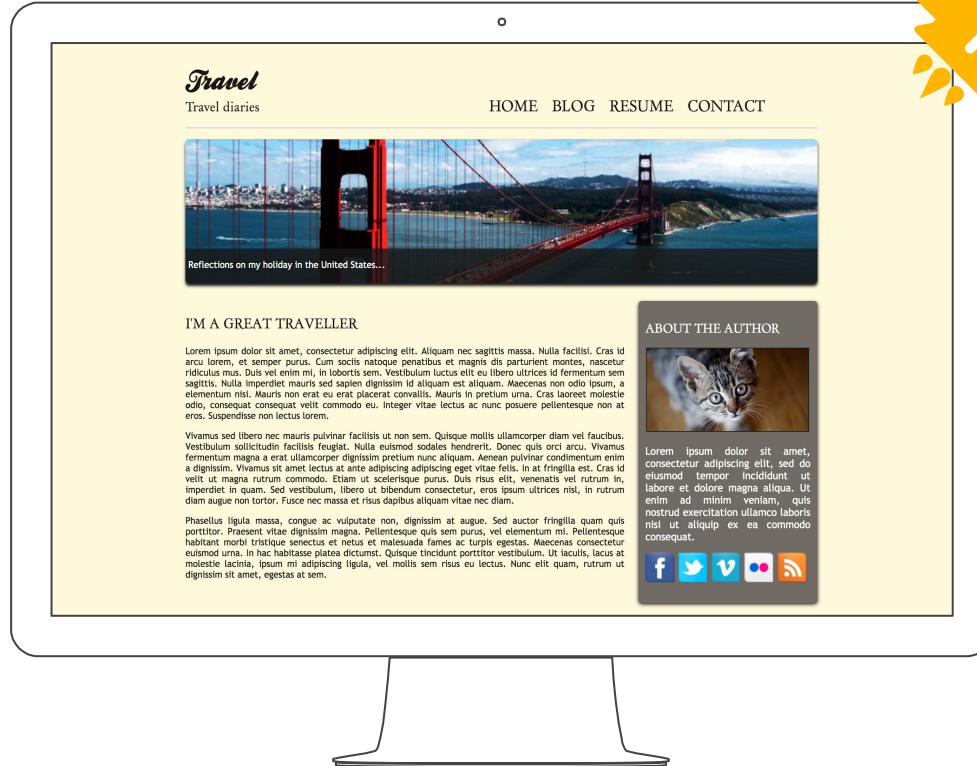
4

Esercizio 1

Traveler

Rifare la pagina usando bootstrap e suoi componenti.

<https://raw.githubusercontent.com/howtosrc/frontend-padova-21/master/es/es1.zip?token=ABCQIKV7Q6F2QZWTWZDF4XLAY5K6K>





Usare font da file

```
@font-face {  
    font-family: 'BallparkWeiner';  
    src: url('fonts/ballpark.eot');  
    src: url('fonts/ballpark.eot?#iefix') format('embedded-opentype'),  
        url('fonts/ballpark.ttf') format('truetype'),  
        url('fonts/ballpark.svg#BallparkWeiner') format('svg');  
    font-weight: normal;  
    font-style: normal;  
}
```

I font sono contenuti dentro la cartella **fonts/** nella root del progetto.

5

Esercizio 2



JANE BLOGLIFE

Welcome to the blog of **Jane's world**



6

Flexbox

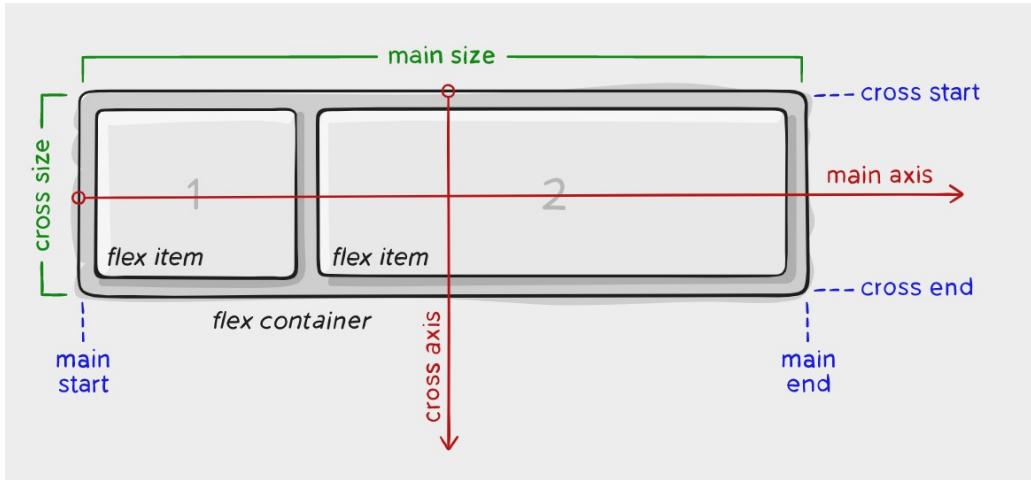


Flex box

- Flexbox è un modello di layout. Nasce con l'obiettivo di creare un modo efficiente per disporre, allineare e distribuire gli elementi in un container anche quando la loro dimensione è sconosciuta o dinamica.
- L'idea è dare al container la possibilità di modificare altezza e larghezza dei suoi figli.



Flex box



main axis

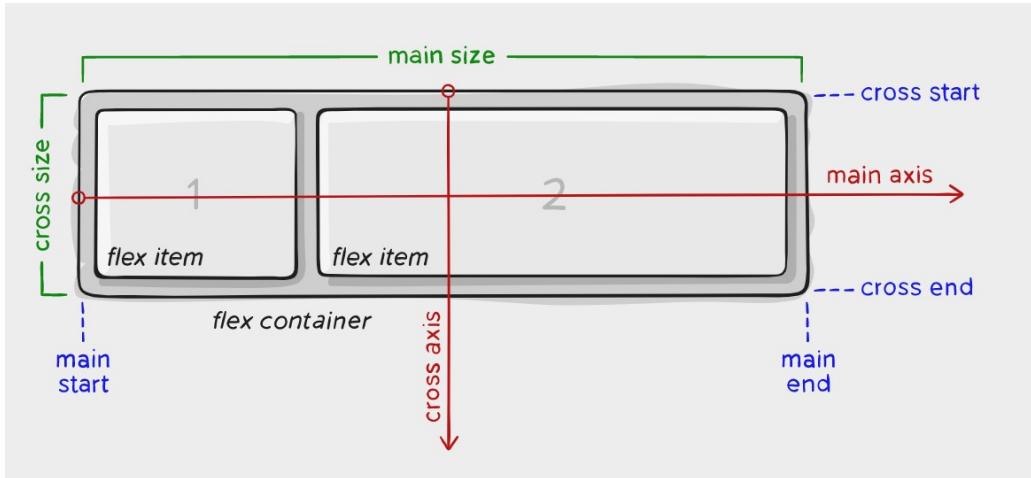
L'asse principale lungo la quale gli elementi vengono disposti

main-start | main-end

Gli elementi sono piazzati nel container partendo da «main-start» e arrivando fino a «main-end»



Flex box



main size

La dimensione (width, height) che corrispondente al main axis

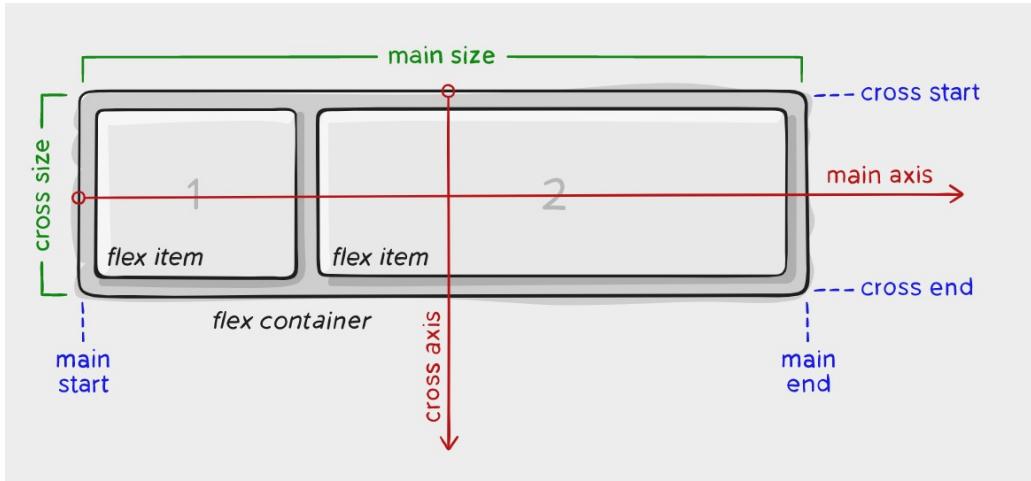
- Height => asse X
- Width => asse Y

cross axis

L'asse perpendicolare alla main axis



Flex box



cross-start | cross-end

Gli elementi sono piazzati a partire da cross-start e fino a cross-end

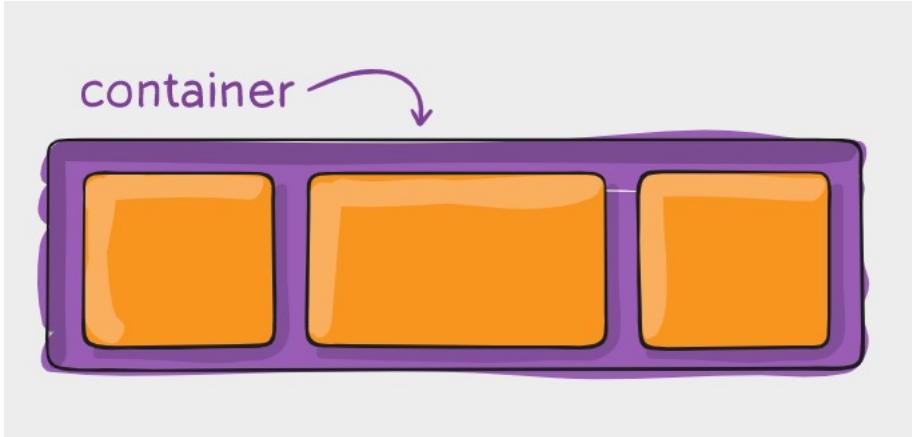
cross size

L'asse perpendicolare alla main axis

Flexbox: regole per il conrainer



Container



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

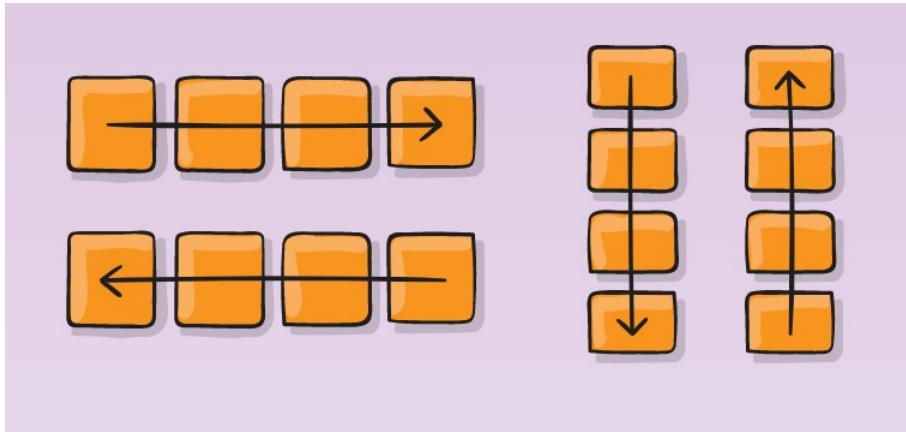
Il container è l'elemento che contiene gli elementi.

Ad esso si applica la proprietà **flex** (flex o inline-flex)

Si applica al container.



Flex-direction



La proprietà **flex-direction** definisce il **main-axis** e di conseguenza tutte le altre proprietà.

Si applica al container.

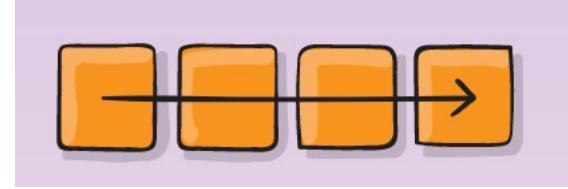
```
.container {  
  flex-direction: row | row-reverse | column | column-reverse;  
}
```



Flex-direction

```
.container {  
    flex-direction: row;  
}
```

row => Figli disposti da sinistra a destra (**ltr**)

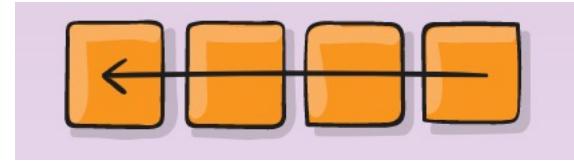




Flex-direction

```
.container {  
    flex-direction: row-reverse;  
}
```

row-reverse => Figli
disposti da destra a
sinistra a destra (**rtf**)

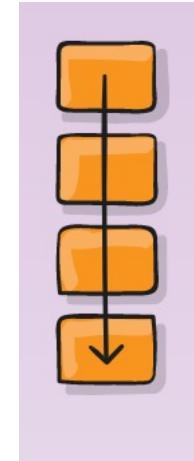




Flex-direction

```
.container {  
    flex-direction: column;  
}
```

column => Figli disposti
dall'altro verso il basso

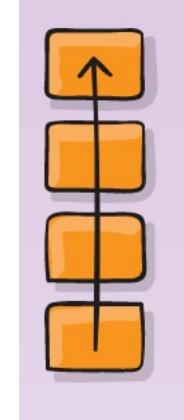




Flex-direction

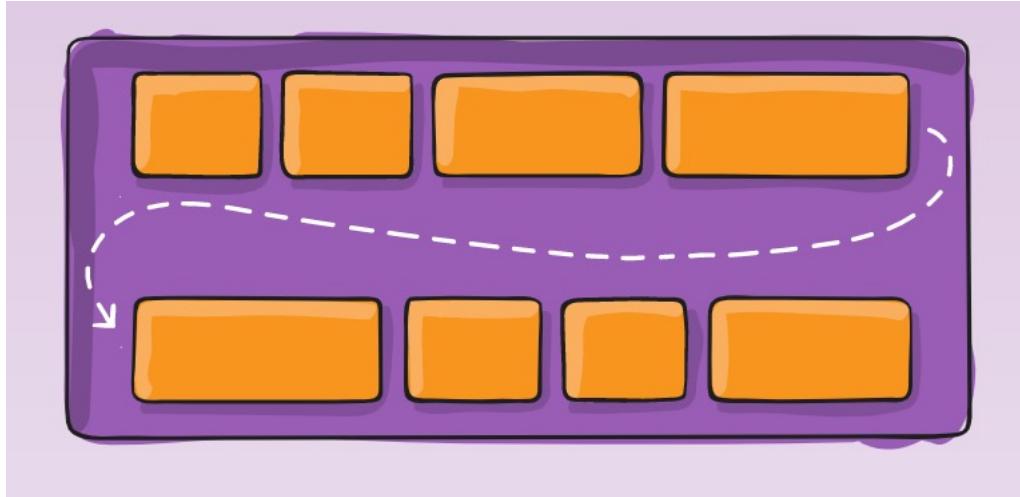
```
.container {  
    flex-direction: column-reverse;  
}
```

column-reverse => Figli
disposti dal basso verso
l'alto





Flex-wrap



```
.container {  
  flex-wrap: nowrap | wrap | wrap-reverse;  
}
```

Di default flex prova a disporre gli elementi lungo la riga,
modificandone le dimensioni in base alle necessità

E' possibile variare questo comportamento con flex-wrap



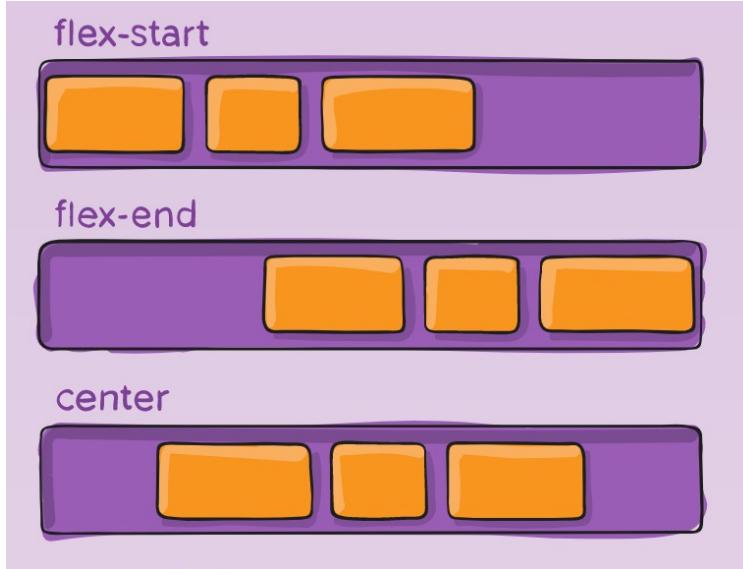
Flex-wrap

```
.container {  
    flex-wrap: nowrap | wrap | wrap-reverse;  
}
```

- **nowrap** => tutti gli elementi piazzati in una sola riga
- **wrap** => disponi gli elementi in diverse linee (dall'alto verso il basso)
- **wrap-reverse** => disponi gli elementi in diverse linee (dal basso verso l'alto)



Justify-content



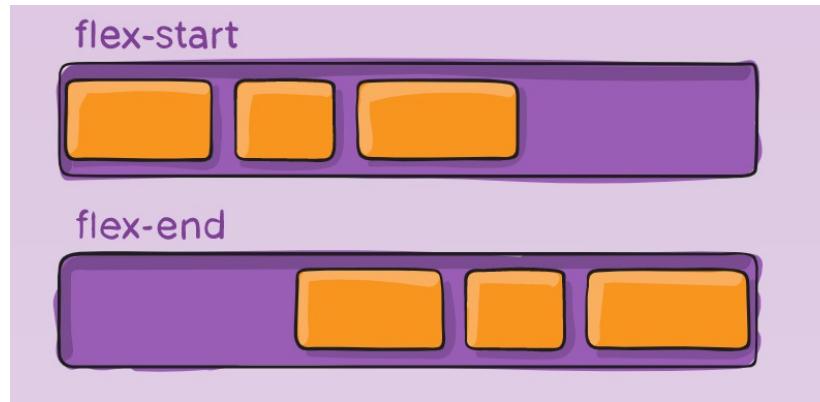
Definisce l'allineamento
lungo la **main-axis**

```
.container {  
  justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly  
}
```



Justify-content

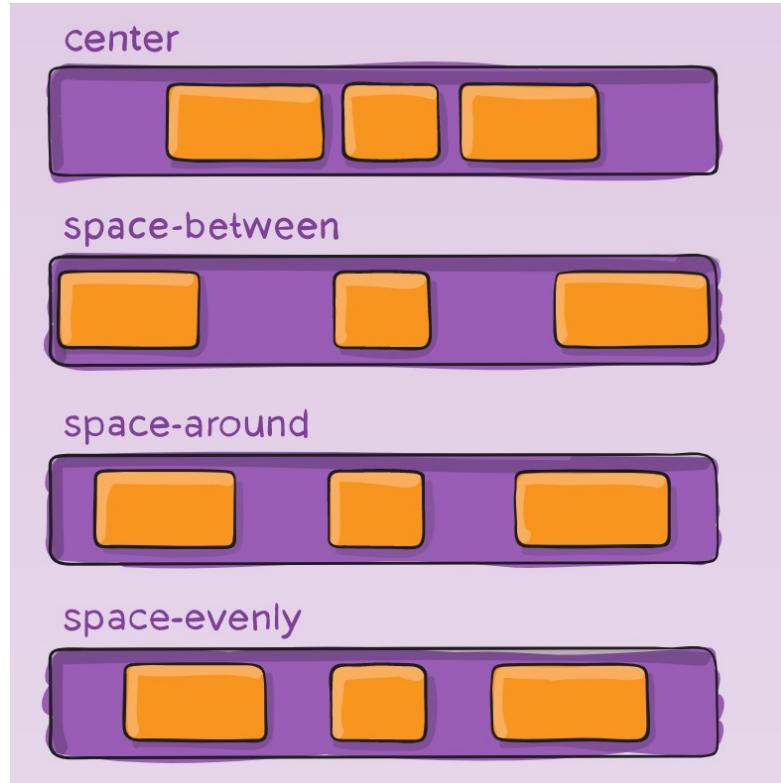
```
.container {  
    flex-direction: flex-start | flex-end;  
}
```





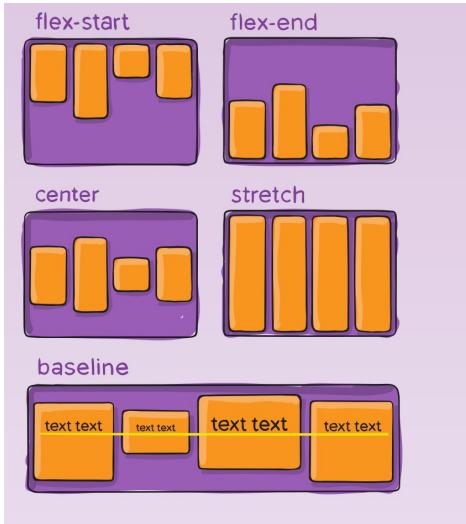
Justify-content

```
.container {  
    flex-direction: center |  
    space-between | space-around |  
    space-evenly;  
}
```





Align-items



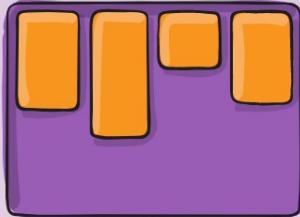
Definisce l'allineamento
lungo la **cross-axis**

```
.container {  
  align-items: stretch | flex-start | flex-end | center | baseline;  
}
```

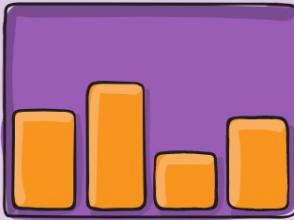


Align-items

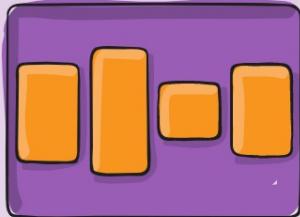
flex-start



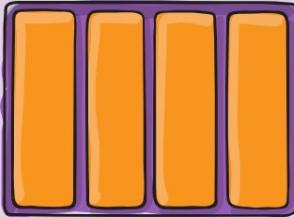
flex-end



center

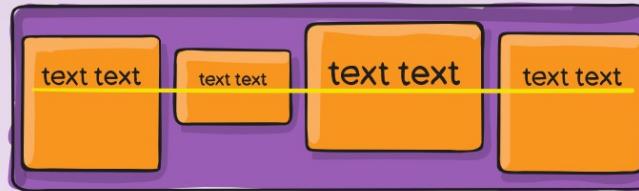


stretch



baseline

baseline



Esercizio 3





Linda Hart
6765 Shady Ln Dr



Roger Medina
2420 Paddock Way



Dean Sullivan
6727 Airplane Ave

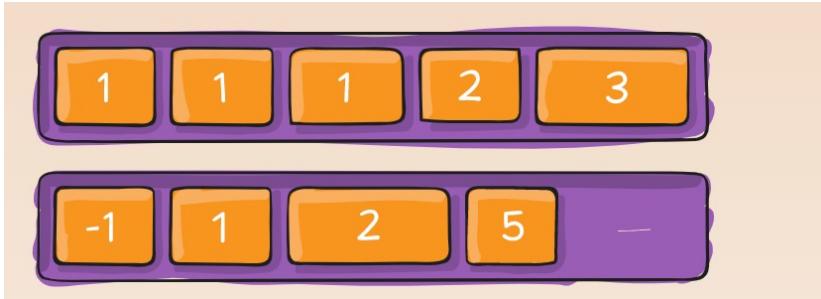


Elizabeth Austin
6685 Blossom Hill Rd

Ricreare il layout qui a fianco senza usare bootstrap e/o float.
Usare flex e css
«custom»

Flexbox: regole per i figli

Order



order permette di cambiare l'ordine in cui gli elementi sono visualizzati. E' una regola che si applica ai figli di un container flex.

Di default gli elementi sono disposti nell'ordine definito nel sorgente della pagina.

Con **order** possiamo modificare questo ordine.

Order



```
<div class="box">  
    <div><a href="#">1</a></div>  
    <div><a href="#">2</a></div>  
    <div><a href="#">3</a></div>  
    <div><a href="#">4</a></div>  
    <div><a href="#">5</a></div>  
</div>
```

```
.box {  
    display: flex;  
    flex-direction: row;  
}
```

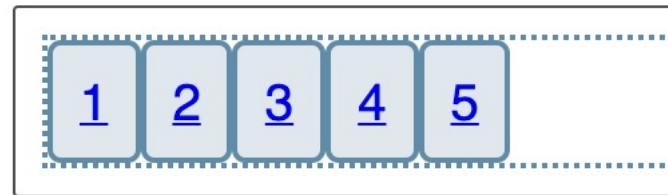
order permette di cambiare l'ordine in cui gli elementi sono visualizzati. E' una regola che si applica ai figli di un container flex.



Order

```
<div class="box">  
  <div><a href="#">1</a></div>  
  <div><a href="#">2</a></div>  
  <div><a href="#">3</a></div>  
  <div><a href="#">4</a></div>  
  <div><a href="#">5</a></div>  
</div>
```

```
.box {  
  display: flex;  
  flex-direction: row;  
}
```





Order

```
<div class="box">  
    <div><a href="#">1</a></div>  
    <div><a href="#">2</a></div>  
    <div><a href="#">3</a></div>  
    <div><a href="#">4</a></div>  
    <div><a href="#">5</a></div>  
</div>
```

```
.box {  
    display: flex;  
    flex-direction: row;  
}  
  
.box :nth-child(1) { order: 2; }  
.box :nth-child(2) { order: 3; }  
.box :nth-child(3) { order: 1; }  
.box :nth-child(4) { order: 3; }  
.box :nth-child(5) { order: 1; }
```



Order

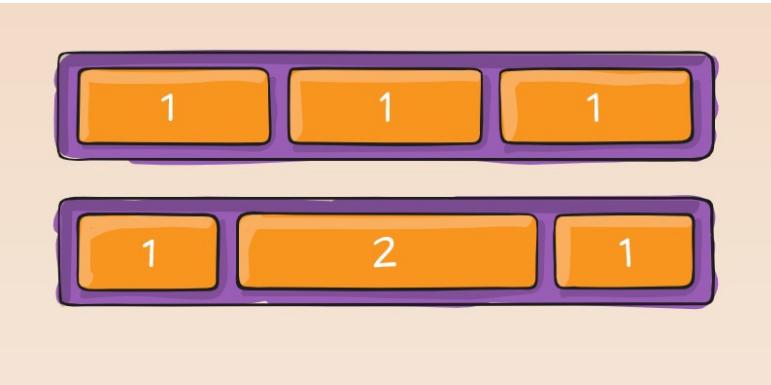
```
<div class="box">  
  <div><a href="#">1</a></div>  
  <div><a href="#">2</a></div>  
  <div><a href="#">3</a></div>  
  <div><a href="#">4</a></div>  
  <div><a href="#">5</a></div>  
</div>
```

```
.box {  
  display: flex;  
  flex-direction: row;  
}  
  
.box :nth-child(1) { order: 2; }  
.box :nth-child(2) { order: 3; }  
.box :nth-child(3) { order: 1; }  
.box :nth-child(4) { order: 3; }  
.box :nth-child(5) { order: 1; }
```





Flex-grow



Di default gli elementi non crescono

Ovvero, di default, **flex-grow è 0**.

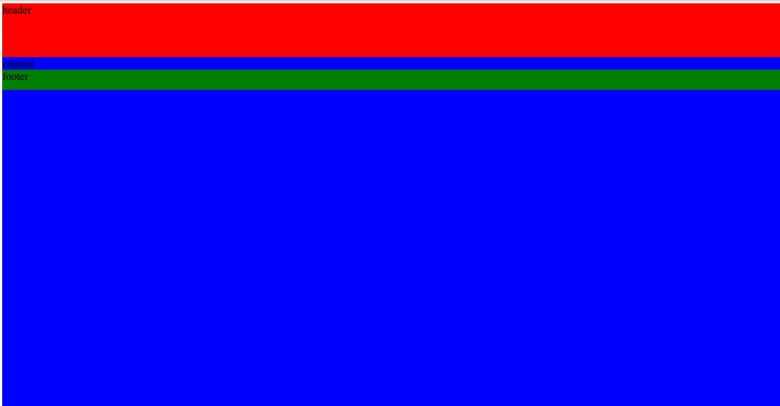
flex-grow definisce l'abilità dei figli di **crescere se necessario**.

Accetta un valore senza unità di misura ed indica una proporzione: quanto, dello spazio rimanente, l'elemento deve occupare



Flex-grow

```
<div class="container">  
  <div class="header">header</div>  
  <div class="content">content</div>  
  <div class="footer">footer</div>  
</div>
```



```
.container {  
  display: flex;  
  flex-direction: column;  
  height: 600px;  
  background: blue;  
}  
  
.header {  
  height: 80px;  
  background: red;  
}  
  
.footer {  
  height: 30px;  
  background: green;  
}
```

Flex-grow

```
<div class="container">
  <div class="header">header</div>
  <div class="content">content</div>
  <div class="footer">footer</div>
</div>
```



```
.container {
  display: flex;
  flex-direction: column;
  height: 600px;
  background: blue;
}

.header {
  height: 80px;
  background: red;
}

.footer {
  height: 30px;
  background: green;
}

.content {
  flex-grow: 1;
}
```





Flex-shrink

Flex-shrink definisce l'abilità dei figli di **contrarsi se necessario**.

Accetta un valore senza unità di misura ed indica una proporzione.

Di default gli elementi si contraggono se necessario.

Ovvero, di default, **flex-shrink è 1**.



Flex-shrink

```
<div class="container">  
  <p class="child">1</p>  
  <p class="child">2</p>  
  <p class="child">3</p>  
</div>
```

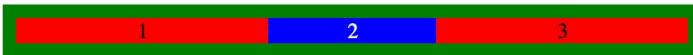


```
.container {  
  display: flex;  
  flex-direction: row;  
  width: 500px;  
  
  background: green;  
  padding: 10px;  
}  
  
.child {  
  width: 250px;  
  text-align: center;  
  margin: 0;  
  background: red;  
}  
  
.child:nth-child(2n) {  
  background: blue;  
  color: white;  
}
```



Flex-shrink

```
<div class="container">  
  <p class="child">1</p>  
  <p class="child">2</p>  
  <p class="child">3</p>  
</div>
```

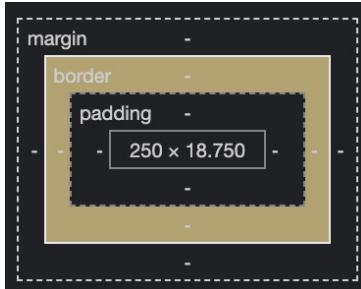
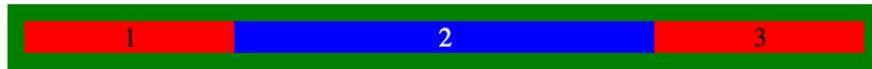


```
.container {  
  display: flex;  
  flex-direction: row;  
  width: 500px;  
  
  background: green;  
  padding: 10px;  
}  
  
.child {  
  width: 250px;  
  text-align: center;  
  margin: 0;  
  background: red;  
}  
  
.child:nth-child(2n) {  
  background: blue;  
  color: white;  
  
  flex-shrink: 2;  
}
```



Flex-shrink

```
<div class="container">  
  <p class="child">1</p>  
  <p class="child">2</p>  
  <p class="child">3</p>  
</div>
```



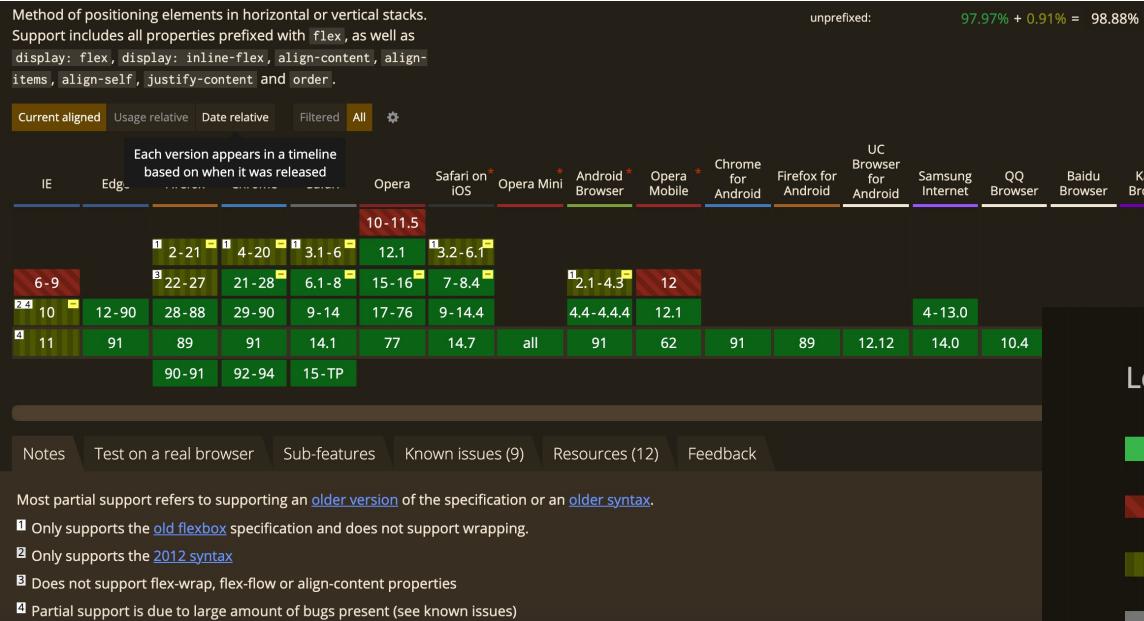
```
.container {  
  display: flex;  
  flex-direction: row;  
  width: 500px;  
  
  background: green;  
  padding: 10px;  
}  
  
.child {  
  width: 250px;  
  text-align: center;  
  margin: 0;  
  background: red;  
}  
  
.child:nth-child(2n) {  
  background: blue;  
  color: white;  
  
  flex-shrink: 0;  
}
```

10

Compatibilità con i browser



Compatibilità con i browser





Compatibilità con i browser

Non tutti i browser supportano tutte le regole CSS.

E' necessario quindi verificare la compatibilità delle regole che stiamo utilizzando con tutti i browser per i quali il progetto è sviluppato.

Sarà quindi necessario fare un **tradeoff** tra l'utilità e facilità di utilizzo di alcune funzionalità (regole css) con il range di browser da supportare.



Compatibilità con i browser

The screenshot shows the homepage of Can I use. At the top, there's a navigation bar with 'Home' (highlighted in orange), 'News', 'May 25, 2021 - New feature: COLR/CPAL(v0) Font Formats', 'Compare browsers', and 'About'. Below the navigation is a search bar with 'Can I use' and a 'Settings' button. The main content area has several sections: 'Index of features', 'Latest features' (listing COLR/CPAL(v0) Font Formats, Web Serial API, SVG vector-effect: non-scaling-stroke, CSS -webkit-user-drag property, and CSS Container Queries), 'Most searched features' (listing CSS Grid, Flexbox, WebP image format, gap property for Flexbox, and ES6), 'Test a feature' (with a note about BrowserStack partnership), 'Did you know?' (with a note about voting for features), 'Third party tools' (listing CanIUse Embed and CanIuse Component), and 'Browser scores' (showing Chrome 91: 394 as the current version). A 'Filter features' button is also present.

Can I use è un sito che permette di conoscere la compatibilità di tutte le funzionalità con le varie versioni dei browser.

<https://caniuse.com/>

11

Esercizio

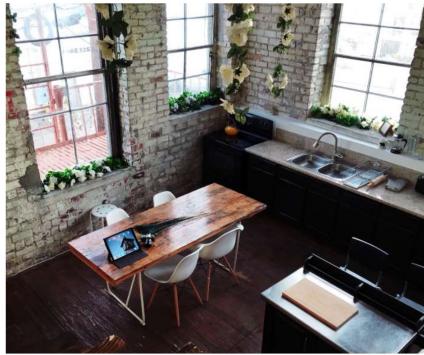


Company
Name

Home
Showcase
Services
Designers
Packages
Contact

Interior Design

Showcase.



12

Esercizio



The image shows a messaging application interface. On the left, a sidebar lists contacts from the TV show Suits, each with a small profile picture and a name. The contacts are: Mike Ross, Louis Litt, Harvey Specter, Rachel Zane, Donna Paulsen, Jessica Pearson, and Harold Gunderson. Below the sidebar is a dark green footer bar with two buttons: 'Add contact' and 'Settings'. The main area is a light gray chat window. At the top, there's a message from Harvey Specter: "Excuses don't win championships." Below it, a message from Mike Ross: "Oh yeah, did Michael Jordan tell you that?". A response from Harvey Specter follows: "No, I told him that.". Another message from Mike Ross asks: "What are your choices when someone puts a gun to your head?". Harvey Specter replies: "What are you talking about? You do what they say or they shoot you.". Finally, a message from Mike Ross states: "Wrong. You take the gun, or you pull out a bigger one. Or, you call their bluff. Or, you do any one of a hundred and forty six other things." At the bottom of the main window, there's a text input field with the placeholder "Write your message..." and a blue send button.

13

UX: User Experience



UX: User Experience

La **user experience** (UX) comprende tutte le «percezioni e reazioni che derivano dall'uso o dall'aspettativa di utilizzo di un prodotto, sistema o servizio», secondo la definizione proposta dall'[International Organization for Standardization](#) (ISO) secondo cui l'espressione comprende anche «le credenze, le preferenze [...], le risposte fisiche e psicologiche, i comportamenti e risultati» che derivano dall'utilizzo.



UX: User Experience

Di solito la UX viene **associata con lo sviluppo di siti web, applicazioni, etc, in maniera scorretta.**

In realtà riguarda riguarda un **insieme più complesso di elementi che devono lavorare all'unisono per offrire un'esperienza all'utente.**

Es:

- Scoperta del prodotto / la prima volta che si entra in un negozio
- Packaging del prodotto
- etc



UX: User Experience

«**tutto ciò è user experience e tutto ciò incide sull'esperienza con il prodotto**». La user experience riguarda, dunque, «**il modo in cui vivi la tua vita, il servizio in questione e, ovviamente, anche come utilizzi un'applicazione o un sistema informatico**»



UX: User Experience

«Pensa alle diverse fasi di un prodotto o servizio, dalle intenzioni iniziali alle riflessioni finali, dal primo utilizzo ad assistenza, servizio e manutenzione. Devi fare in modo che questi elementi lavorino tutti insieme e con continuità»

Don Norman - psicologo cognitivo, fondatore di Nielsen Norman group ed è considerato il padre dello human design.



UX: User Experience

«Pensa alle diverse fasi di un prodotto o servizio, dalle intenzioni iniziali alle riflessioni finali, dal primo utilizzo ad assistenza, servizio e manutenzione. Devi fare in modo che questi elementi lavorino tutti insieme e con continuità»

Don Norman - psicologo cognitivo, fondatore di Nielsen Norman group ed è considerato il padre dello human design.



UX vs UI: Cos'è User Interface?

La **User Interface** rappresenta l'**interfaccia grafica vera e propria**, quell'insieme di **elementi che permette l'interazione tra l'utente finale e lo strumento digitale**; in un sito web, ad esempio, il pulsante (o *call to action*) è l'elemento dell'interfaccia che consente all'utente di compiere una determinata azione (come quelle elencate precedentemente).



Cattiva UX

Un esempio di cattiva User eXperience è quello di accedere ad un sito e-commerce, trovare a fatica un paio di scarpe da calcio che ci piacciono e non riuscire a capire quali numeri sono disponibili, oppure trovare difficoltà nel processo di registrazione/acquisto perché lungo e complicato.

Quando cerco una cosa devo trovarla il prima possibile, guardando **nel posto dove mi aspetto di trovarla**, principio alla base di tutto il ragionamento, altrimenti il tasso di abbandono aumenta...



Risorse online su UX/UI:

- <https://xd.adobe.com/ideas/process/ui-design/4-golden-rules-ui-design/>
- <https://www.insidemarketing.it/glossario/definizione/user-experience/>
- <https://www.kuma.cloud/blog/user-interface-ui-e-user-experience-ux-qual-e-la-differenza/>



Thanks!

Any questions?

Per qualsiasi domanda contattatemi:
alessandro.pasqualini.1105@gmail.com