**APPUNTI FLEXBOX**

Per prima cosa dobbiamo ricordare il funzionamento di un div container.

Esso, di default, ha una width definita, ed è uguale alla width del div contenitore. Nel caso in cui il container sia il div più esterno del nostro progetto, esso avrà width pari alla width dell’intera pagina web.

Discorso diverso per la height. La height di un div è pari a 0. Mano mano che il div si riempie, la sua height sarà pari alla somma delle height degli elementi che si trovano all’interno del nostro div.

Ovviamente possiamo aumentare la height del nostro container forzando la proprietà height ed impostarla ad un valore di nostro piacimento.

Per prima cosa studieremo le proprietà applicabili al div container. Dopodiché studieremo le proprietà applicabili ai div figli.

**PROPRIETA’ FLEXBOX APPLICABILI AL DIV CONTAINER**

Per il momento non utilizziamo ne la proprietà height ne la proprietà width, ne sul div container ne sui div contenuti all’interno di esso. Consideriamo quindi una situazione iniziale di questo tipo:

HTML

<body>

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

}

p{

    margin:0;

}

#div1{

    border:1px solid black;

    background-color:white;

    font-size: 50px;

    font-weight: bold;

}

#div2{

    border:1px solid black;

    background-color:violet;

    font-size: 50px;

    font-weight: bold;

}

#div3{

    border:1px solid black;

    background-color:blue;

    font-size: 50px;

    font-weight: bold;

}

BROWSER



I div figli contenuti nel div container prendono quindi tutto lo spazio disponibile, ossia prendono tutto lo spazio già occupato dal div container.

Dal codice CSS si evince che il div container ha sfondo orangered, ma dato che i div figli prendono tutto lo spazio disponibile all’interno del div padre, tale sfondo non si vede.

Si vede invece lo sfondo dei div figli: div1 è bianco, div2 è rosa e div3 è blu.

Vediamo ora cosa succede se applichiamo la proprietà display:flex al nostro div container.

HTML

<body>

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

    display:flex;

}

#div1,#div2,#div3{

    font-size: 50px;

    font-weight: bold;

}

p{

    margin:0;

}

#div1{

    border:1px solid black;

    background-color:white;

}

#div2{

    border:1px solid black;

    background-color:violet;

}

#div3{

    border:1px solid black;

    background-color:blue;

}

BROWSER



E’ quindi successo che la lunghezza (width) di ogni div si è ridotta adattandosi al contenuto del div stesso. Dopodiché tutti i div sono stati inseriti sulla stessa riga. Lo spazio non riempito è la parte in rosso, ed è quanto rimane del div container. Guardando il CSS si nota che il div container ha sfondo rosso. Questo significa che la parte in rosso è proprio la parte del div container non occupata dai div figli.

C’è un modo per far sparire la parte in rosso? Certo. Basta modificare la proprietà display:flex in display:inline-flex.

HTML

<body>

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

    display:inline-flex;

}

#div1,#div2,#div3{

    font-size: 50px;

    font-weight: bold;

}

#div1{

    border:1px solid black;

    background-color:white;

}

#div2{

    border:1px solid black;

    background-color:violet;

}

#div3{

    border:1px solid black;

    background-color:blue;

}

BROWSER



**FLEX DIRECTION PROPERTY**

Torniamo ora indietro, al punto in cui avevamo display:flex

HTML

<body>

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

    display:flex;

}

#div1,#div2,#div3{

    font-size: 50px;

    font-weight: bold;

}

#div1{

    border:1px solid black;

    background-color:white;

}

#div2{

    border:1px solid black;

    background-color:violet;

}

#div3{

    border:1px solid black;

    background-color:blue;

}

BROWSER



Si noti che tutti I div figli sono allineati alla sinistra del div container.

Quanto visto equivale ad aggiungere la proprietà flex-direction:row alla destra al nostro div container. Vediamolo qui sotto.

HTML

<body>

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

    display:flex;

    flex-direction:row;

}

#div1,#div2,#div3{

    font-size: 50px;

    font-weight: bold;

}

#div1{

    border:1px solid black;

    background-color:white;

}

#div2{

    border:1px solid black;

    background-color:violet;

}

#div3{

    border:1px solid black;

    background-color:blue;

}

BROWSER



CVD.

Torniamo ora un secondo alla situazione iniziale, ossia alla situazione in cui non è stato assegnata la proprietà display:flex al nostro container

HTML

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

}

#div1,#div2,#div3{

    font-size: 50px;

    font-weight: bold;

}

#div1{

    border:1px solid black;

    background-color:white;

}

#div2{

    border:1px solid black;

    background-color:violet;

}

#div3{

    border:1px solid black;

    background-color:blue;

}

BROWSER

Immagine che contiene testo, schermata, Carattere, Policromia

Descrizione generata automaticamente

Questa situazione di base equivale ad utilizzare display:flex e display-direction:column al nostro container:

HTML

<body>

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

    display:flex;

    flex-direction:column;

}

#div1,#div2,#div3{

    font-size: 50px;

    font-weight: bold;

}

#div1{

    border:1px solid black;

    background-color:white;

}

#div2{

    border:1px solid black;

    background-color:violet;

}

#div3{

    border:1px solid black;

    background-color:blue;

}

BROWSER

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

CVD

Torniamo alla situazione alla quale abbiamo display:flex per il nostro div container (Che ricordiamo coincide con la situazione in cui abbiamo display:flex e flex-direction:row)

HTML

<body>

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

    display:flex;

}

#div1,#div2,#div3{

    font-size: 50px;

    font-weight: bold;

}

#div1{

    border:1px solid black;

    background-color:white;

}

#div2{

    border:1px solid black;

    background-color:violet;

}

#div3{

    border:1px solid black;

    background-color:blue;

}

BROWSER



In questo caso I nostri div figli sono allineati da sinistra a destra del nostro div container.

E’ possibile allinearli, al contrario, da destra a sinistra? Certo! Si aggiunge al nostro div container la proprietà flex-direction: row-reverse; al nostro div padre!

HTML

<body>

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

    display:flex;

    flex-direction: row-reverse;

}

#div1,#div2,#div3{

    font-size: 50px;

    font-weight: bold;

}

#div1{

    border:1px solid black;

    background-color:white;

}

#div2{

    border:1px solid black;

    background-color:violet;

}

#div3{

    border:1px solid black;

    background-color:blue;

}

BROWSER



In questo caso i div si sono ribaltati: Il primo div da sinistra diventa il primo da destra (quindi va per ultimo), il secondo div da sinistra diventa il secondo da destra (quindi resta al centro) ed il terzo div da sinistra diventa il terzo da destra (Quindi diventa il primo).

Torniamo all’esempio in cui abbiamo display:flex e display-direciton:column. Ricordiamo che l’inserimento di queste due proprietà corrisponde alla situazione in cui non abbiamo inserito la proprietà display:flex, in quanto di div sono inseriti per colonna. Ricordiamo che la situazione era la seguente.

Immagine che contiene testo, schermata, Carattere, Policromia

Descrizione generata automaticamente

Anche in questo caso, tuttavia, possiamo ribaltare i div: Il primo diventa l’ultimo, il centrale resta tale e l’ultimo diventa il primo. Per far ciò, dobbiamo utilizzare la solita proprietà display:flex; sul div padre ed aggiungere, sempre sul div padre, la proprietà flex-direction:column-reverse

HTML

<body>

    <div id="container">

        <div id="div1">SEI BRUTTO</div>

        <div id="div2">SEI BELLO</div>

        <div id="div3">FAI SCHIFO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    background-color:orangered;

    display:flex;

    flex-direction: column-reverse;

}

#div1,#div2,#div3{

    font-size: 50px;

    font-weight: bold;

}

#div1{

    border:1px solid black;

    background-color:white;

}

#div2{

    border:1px solid black;

    background-color:violet;

}

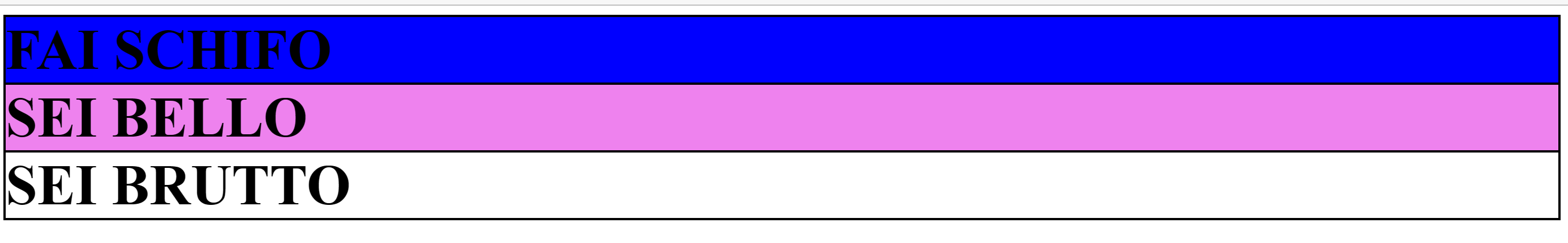
#div3{

    border:1px solid black;

    background-color:blue;

}

BROWSER



CVD

Negli esempi visti finora non abbiamo considerato un div container con una width e una heght scelti dall’utente. Vediamo il seguente esempio.

HTML

<body>

    <div id="container">

        <div>CANE</div>

        <div>GATTO</div>

        <div>CANARINO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

}

div{

    height:20%;

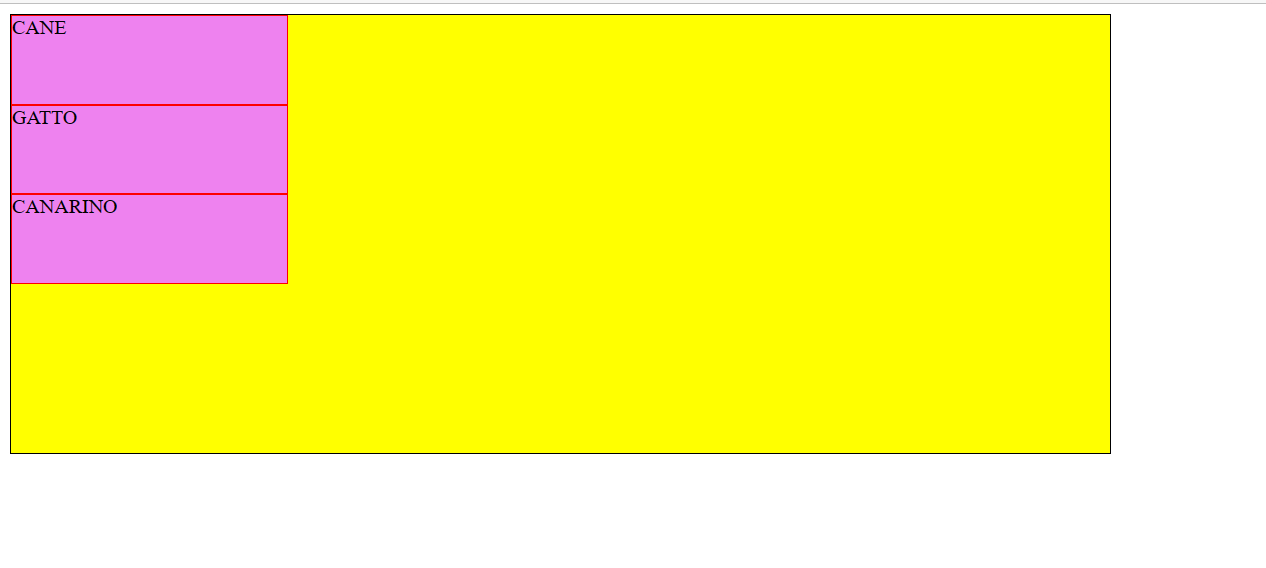
    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Ovviamente, come al solito, possiamo applicare display:flex al nostro div container, e ricordiamo che questa proprietà sottointende la proprietà flex-direction:column. Vediamo tutte le proprietà flex-direction applicate al nostro container.

Flex-direction: column

CSS  
#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:column;

}

div{

    height:20%;

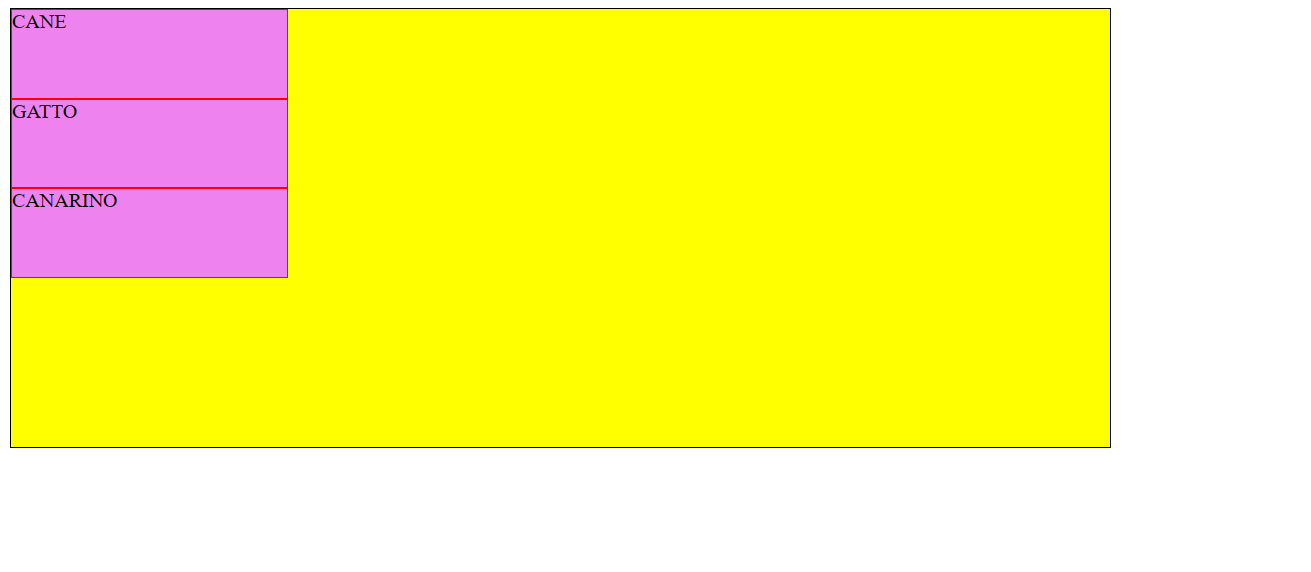
    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Flex-direction:row

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:row;

}

div{

    height:20%;

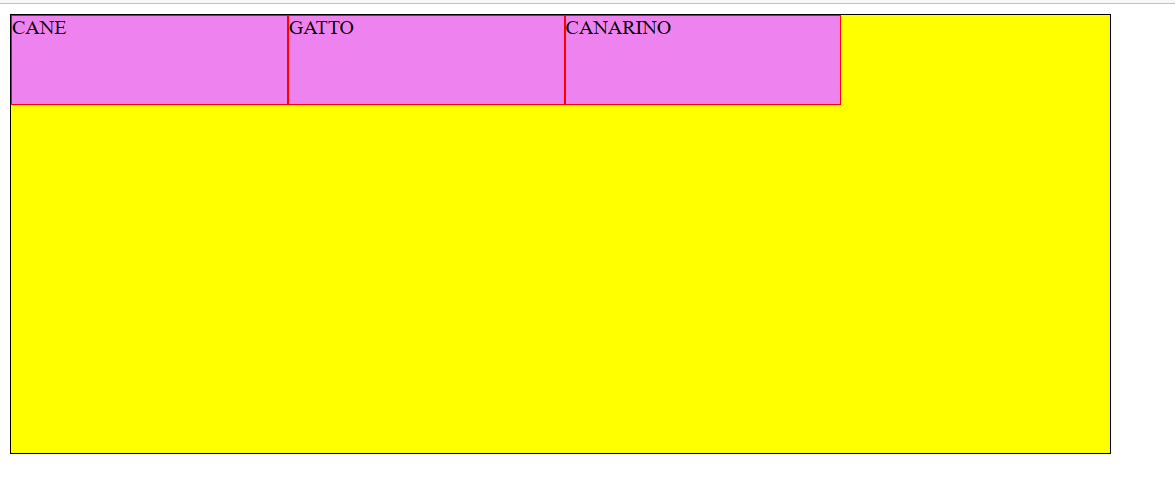
    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



L’esempio appena applicato ci mostra il flex-direction:row nel caso in cui abbiamo applicato la proprietà width e height ai nostri div figli del container. Si noti, infatti, che l’altezza del nostro div container è pari a 351.6px. L’altezza dei div figli è il 20% dell’altezza del container, e quindi è pari a 71.6px circa (aprire il file in microsoft edge e usare l’inspector per essere sicuri di quello che ho scritto). Vediamo cosa succede se togliamo la proprietà height ai figli (Ma non al container):

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:row;

}

div{

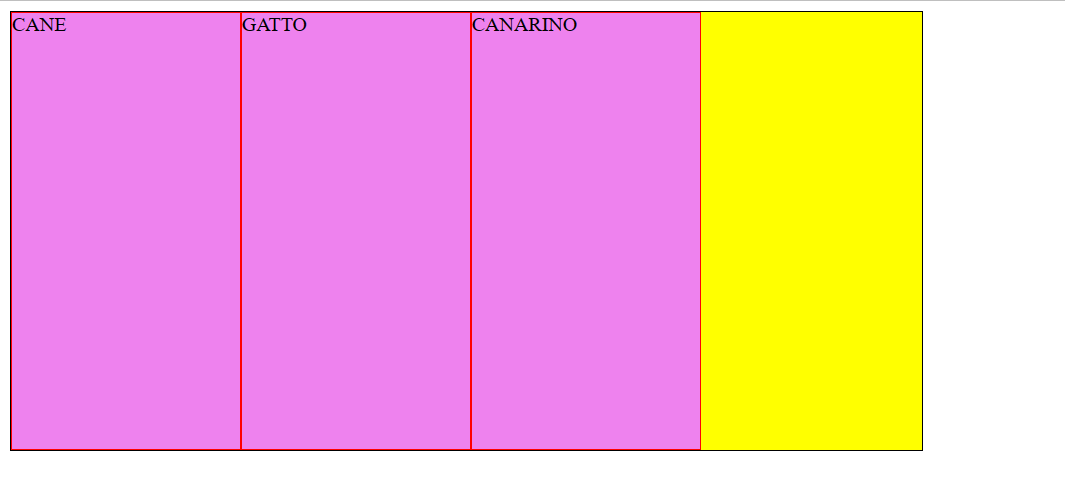
    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



L’altezza dei div figli si adatta all’altezza del div container!!

Passiamo ora alla larghezza. Da codice, la larghezza dei div figli è pari al 25% della larghezza del div container.

Aprendo l’inspector, si nota che la parte rimanente del container è quasi uguale alla larghezza del singolo div. Se togliamo la proprietà width dal nostro css, noteremo che la larghezza dei div si adatta alla larghezza del contenuto del div stesso:

CSS  
#container{

    border:1px solid black;

    width:60%;

    height:350px;

    box-sizing: border-box;

    background-color:yellow;

    display:flex;

    flex-direction:row;

}

div{

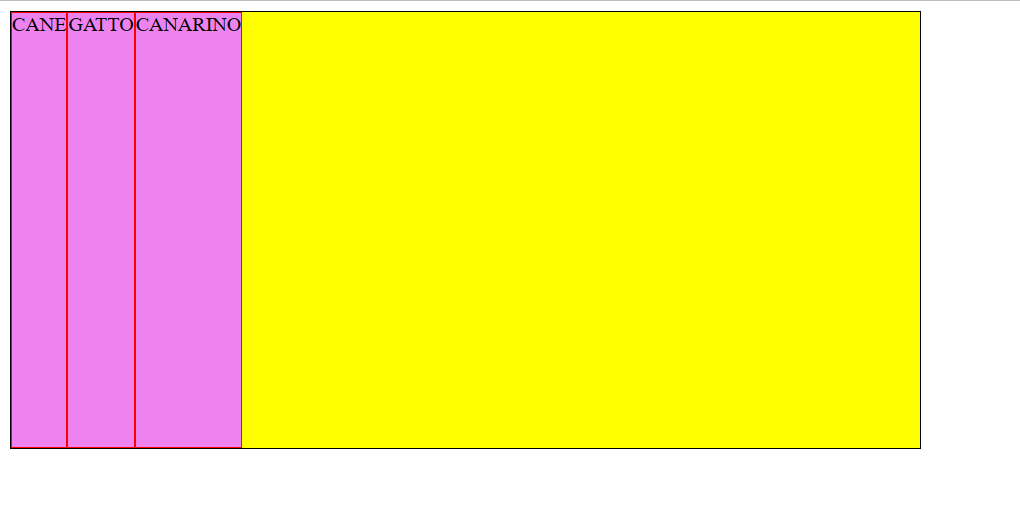
    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



CVD.

Torniamo alla nostra situazione iniziale.

HTML

<body>

    <div id="container">

        <div>CANE</div>

        <div>GATTO</div>

        <div>CANARINO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:row;

}

div{

    height:20%;

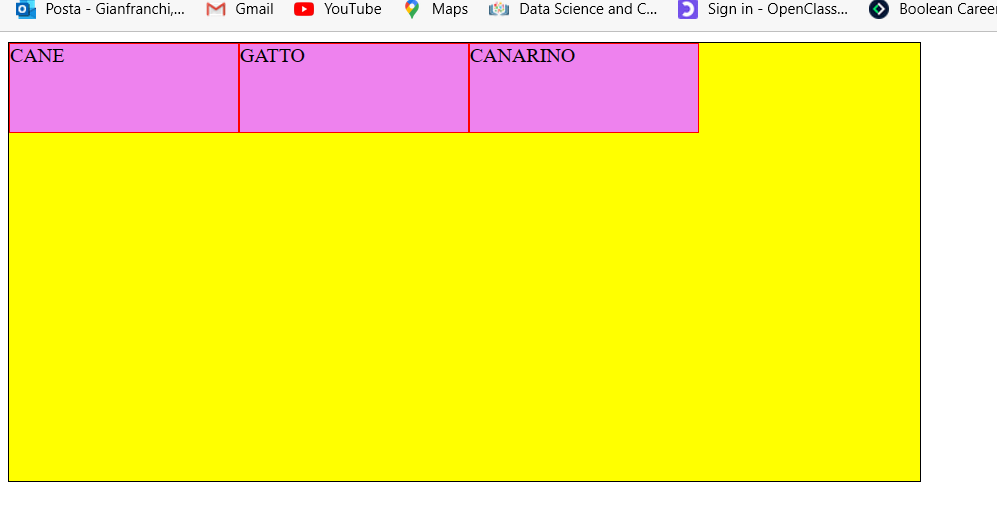
    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Torniamo a vedere i nostri flex-direction. Ricordiamo che la proprietà appena vista è flex-direction: column

flex-direction: row-reverse

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:row-reverse;

}

div{

    height:20%;

    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Flex-direction:column-reverse

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:column-reverse;

}

div{

    height:20%;

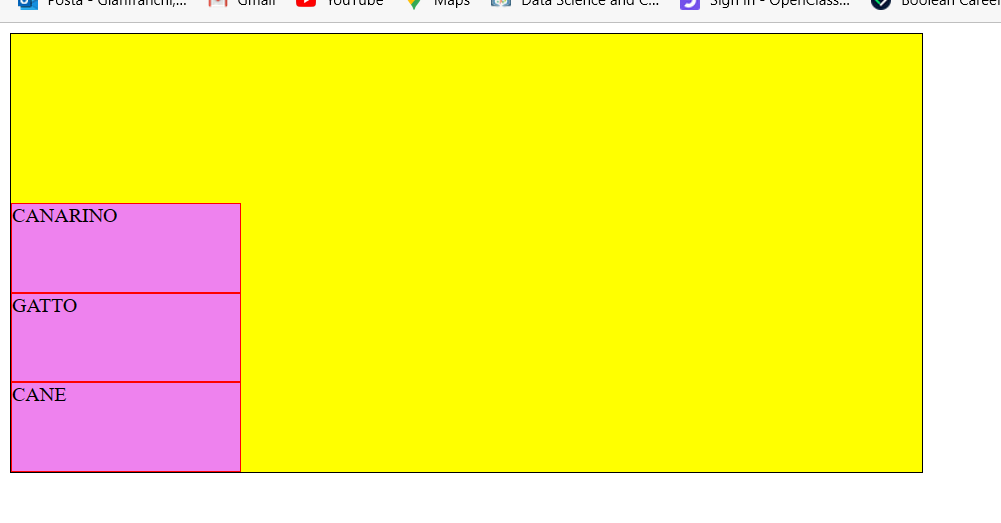
    width:25%;

    border:1px solid red;

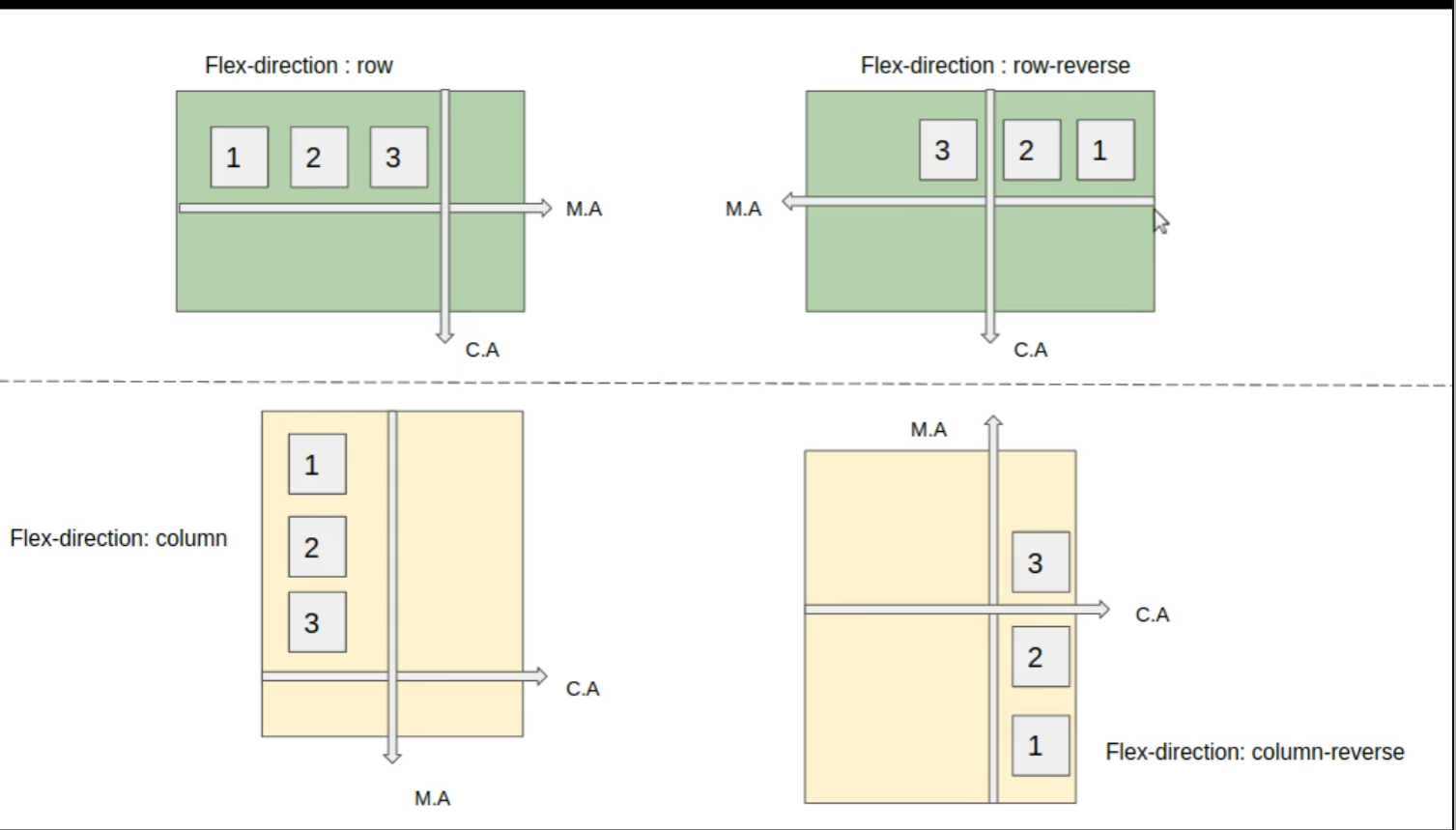
    background-color:violet;

}

BROWSER



Quanto visto finora può essere riassunto dal seguente grafico.



**SPOSTAMENTO LUNGO L’ASSE DELLE X DEI DIV FIGLI ALL’INTERNO DEL DIV CONTAINER TRAMITE LA PROPRIETA JUSTIFY-CONTENT.**

Torniamo nella nostra situazione iniziale.

HTML

<body>

    <div id="container">

        <div>CANE</div>

        <div>GATTO</div>

        <div>CANARINO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

}

div{

    height:20%;

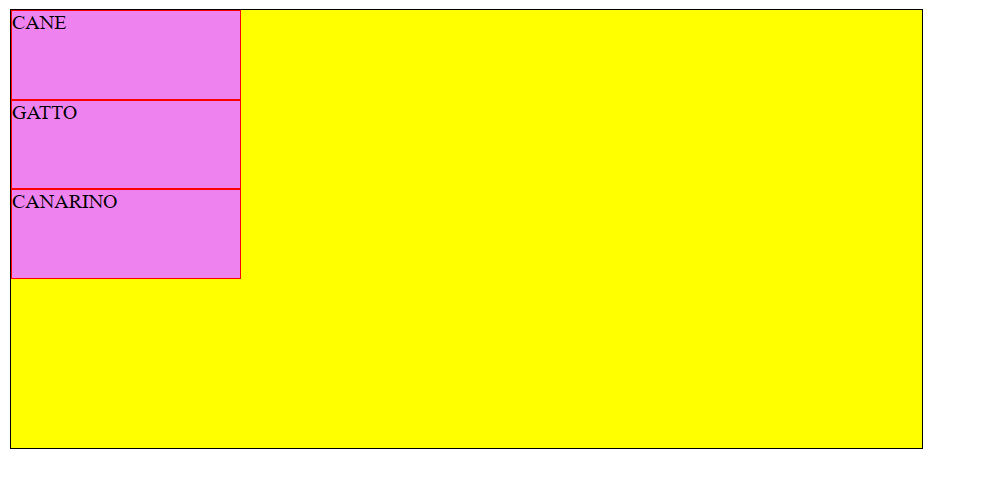
    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Utilizziamo ora la proprietà justify-content, combinata con la proprietà display:flex, per spostare i nostri elementi figli all’interno del div padre.

Si noti che, di default, il nostro div container ha la proprietà justify-content valorizzata a flex-start.

Justify-content può assumere I seguenti valori

* flex-start
* flex-end.
* center
* space-between.

Come detto, di default justify-content è valorizzato a flex-start.

Quindi, di default, il nostro div container avrà le seguenti proprietà:

display:flex

flex-direction: column

justify-content: flex-start

In base al nostro esempio si ha quindi:

HTML

<body>

    <div id="container">

        <div>CANE</div>

        <div>GATTO</div>

        <div>CANARINO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:column;

    justify-content:flex-start;

}

div{

    height:20%;

    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Cambiamo ora il valore della proprietà justify-content per muovere i nostri elementi figli all’interno del container. Per il momento teniamo la proprietà flex-direction fissa al suo valore di default, ossia flex-direction:column.

Justify-content: flex-end



Justify-content:

Modifichiamo ora il valore di flex-direction. Finora abbiamo visto justify-content combinato con il valore column della proprietà flex-direction. Modifichiamo ora il valore di flex-direction.

Flex-direction: row e justify-content:flex-start

CSS  
#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:row;

    justify-content:flex-start;

}

div{

    height:20%;

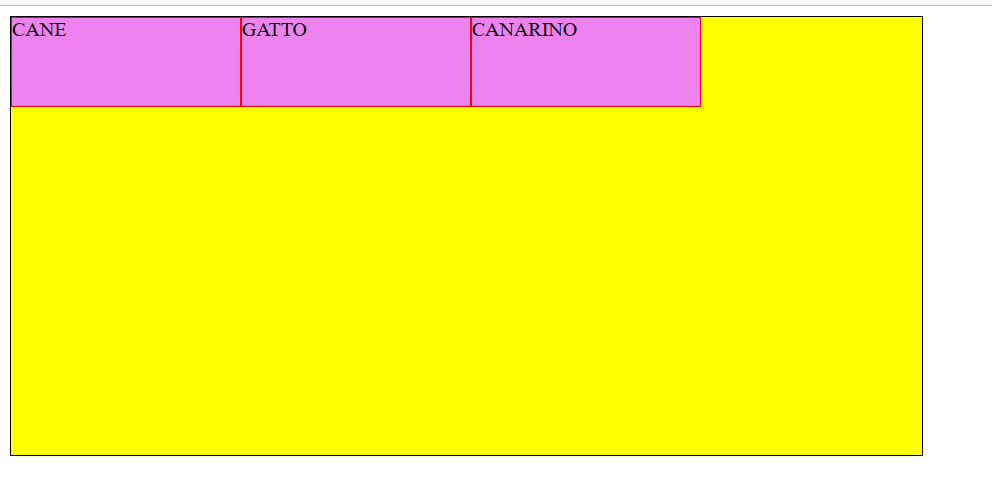
    width:25%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Ovviamente, essendo flex-start il valore di default di jusify-content, l’esempio appena visto corrisponde all’utilizzo di flex-direction: row senza la specifica di justify-content.

Flex-direction: row e justify-content: flex-end:

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:row;

    justify-content:flex-end;

}

div{

    height:20%;

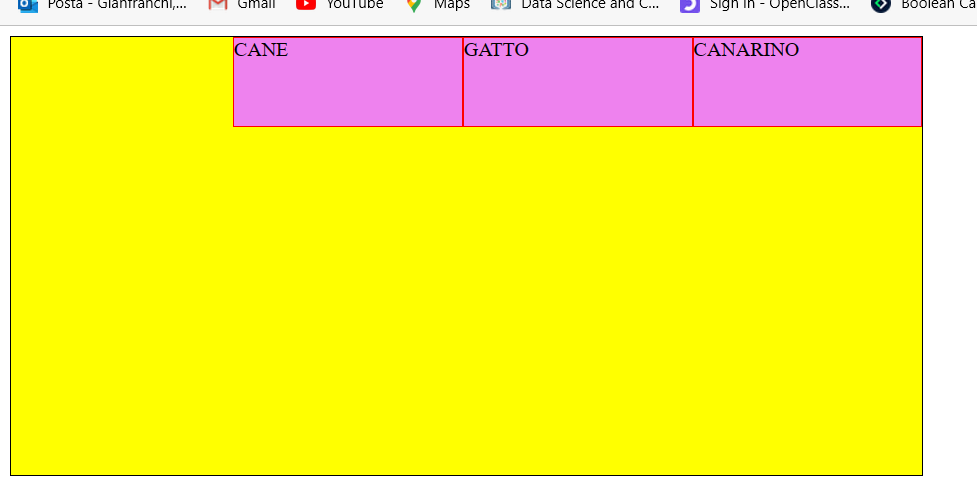
    width:25%;

    border:1px solid red;

    background-color:violet;

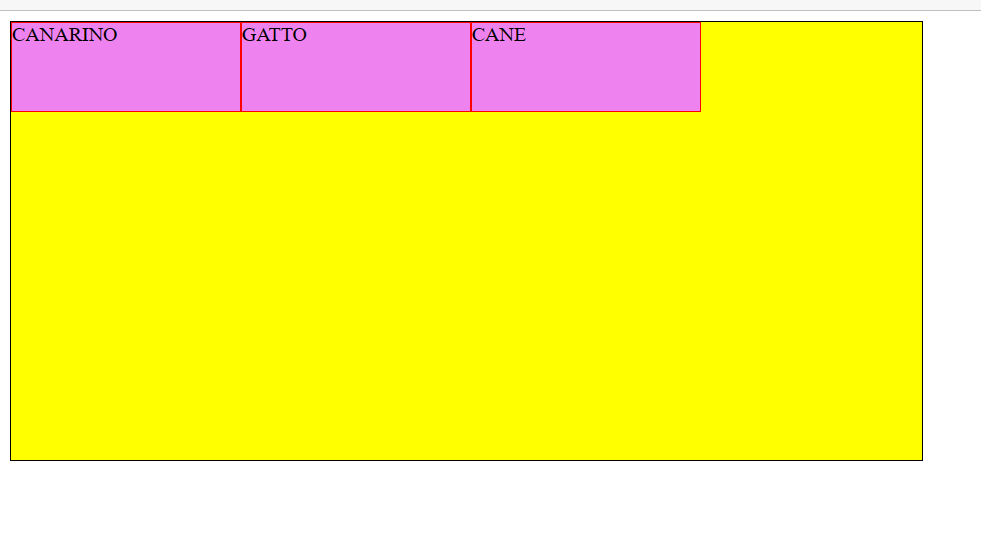
}

BROWSER



Dato che gli esempi con justify-content:flex-start sono già stati visti, utilizziamo solo il justify-content:flex-end combianato con i vari valori di flex-direction:

flex-direction: row-reverse e justify-content: flex-end



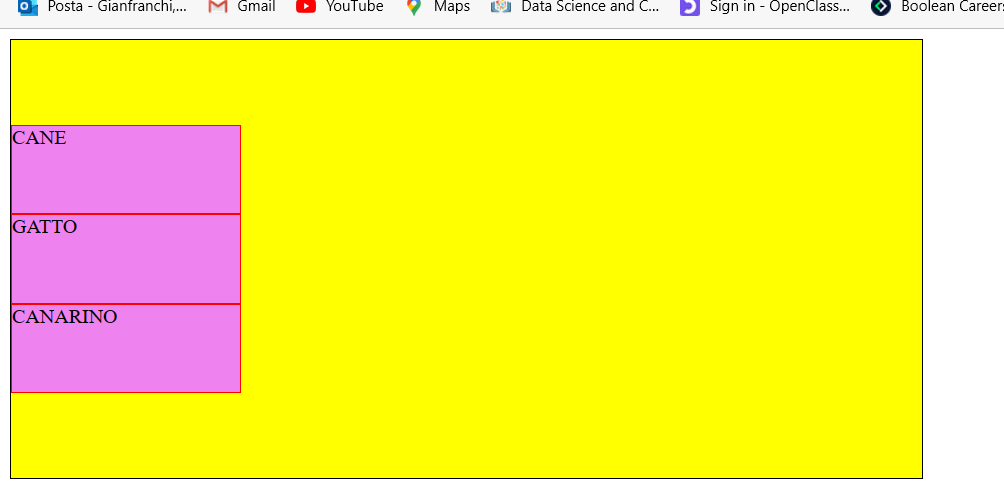
La spiegaizone di quanto visto è facilmente intuibile. Flex-direction:row-reverse sbatte I div figli in fondo al div container in modo inverso. Ciò significa che, se all’inizio abbiamo 3 div messi all’inzio del div container (CANE,GATTO,CANARINO), la proprietà flex-direction:row-reverse sbatte questi div figli in fondo al div container in modo inverso, e quindi abbiamo (CANARINO,GATTO,CANE) in fondo al container. Dopodiché, la proprietà justify-content:flex-end sbatte I div figli dall’altra parte del div container, senza invertirli. Abbiamo così come risultato i nostri div invertiti in precedenza da flex-direction:row-reverse all’inizio del nostro container.

Flex-direction: column-reverse e justify-content:flex-end

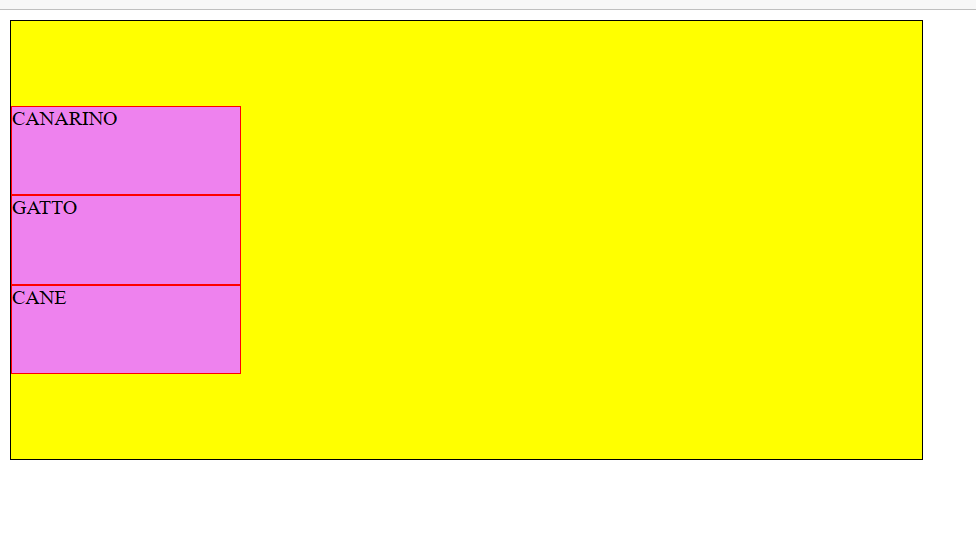


Vediamo ora la combinazione di justify-content:center e i vari flex-direction.

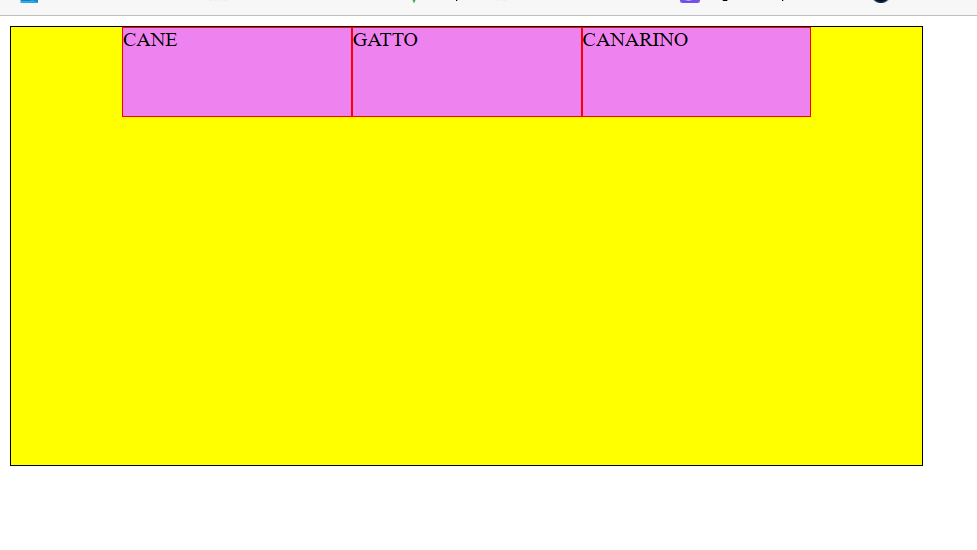
Flex-direction:column e justify-content:center



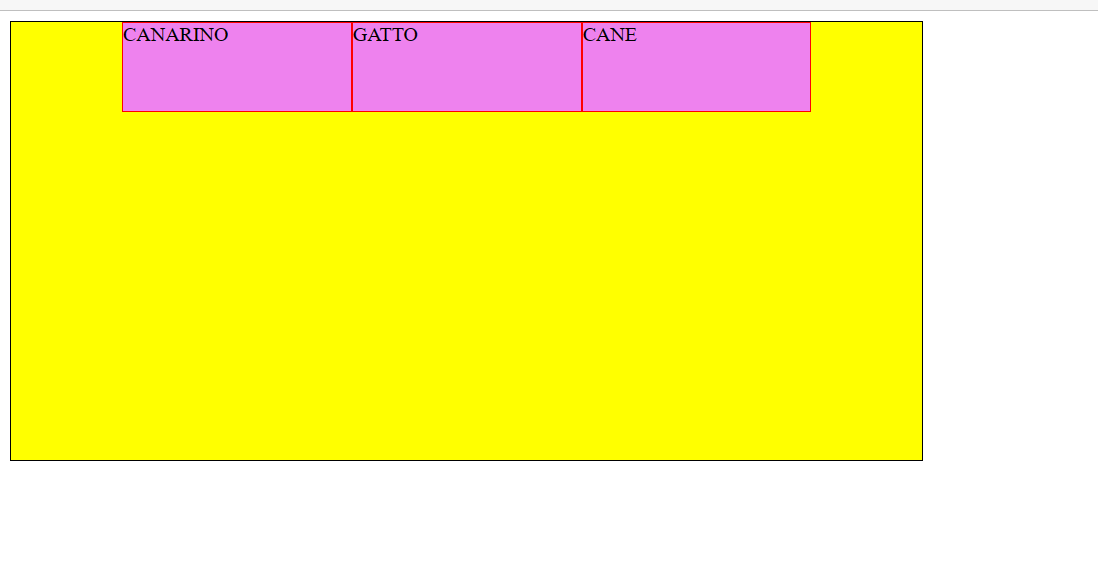
Flex-direction:column-reverse e justify-content:center



Flex-direction:row e justify-content:center



Flex-direction:row-reverse e justify-content:center



Ora che abbiamo capito il meccanismo per quanto riguarda la combinazione tra justify-content e flex-direction, si lascia allo studente il compito di capire come si combinano i vari valori di flex-direction e i seguenti valori di justify-content rimanenti, ossia space-between, space-around etc.

**SPOSTAMENTO LUNGO L’ASSE DELLE Y DEI DIV FIGLI ALL’INTERNO DEL DIV CONTAINER TRAMITE LA PROPRIETA ALIGN-ITEMS.**

Finora abbiamo visto come spostare lungo l’asse delle x gli elementi figli del container, applicando a quest’ultimo la proprietà justify-content. Ora vediamo come spostare lungo l’asse delle y i nostri div figli, applicando al div container la proprietà align-items.

Partiamo dalla situazione iniziale, nella quale tutte le proprietà sono valorizzate al loro valore di default.

* display: flex
* flex-direction:colum
* justify-content: flex-start
* align-items:stretch.

Prendiamo quindi in considerazione il nostro solito esempio:

HTML

<body>

    <div id="container">

        <div>CANE</div>

        <div>GATTO</div>

        <div>CANARINO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:column;

    justify-content:flex-start;

    align-items:stretch;

}

div{

    height:20%;

    width:25%;

    border:1px solid red;

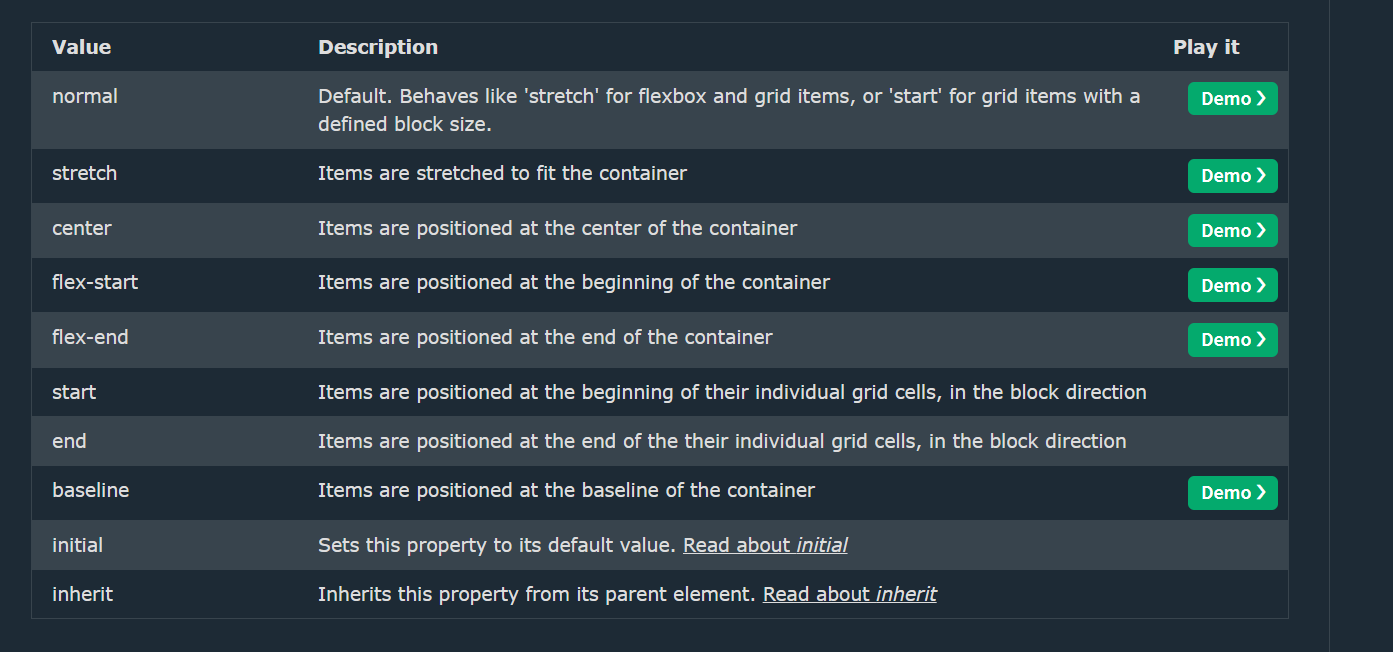
    background-color:violet;

}

BROWSER

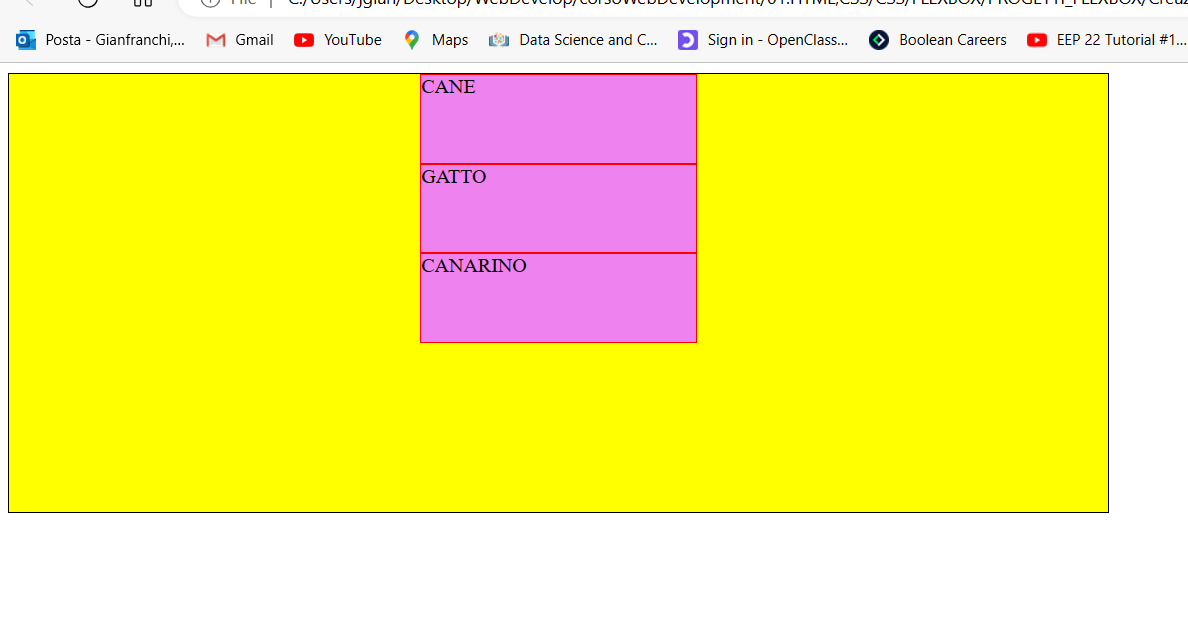


I valori che la proprietà align-items può assumere sono i seguenti:

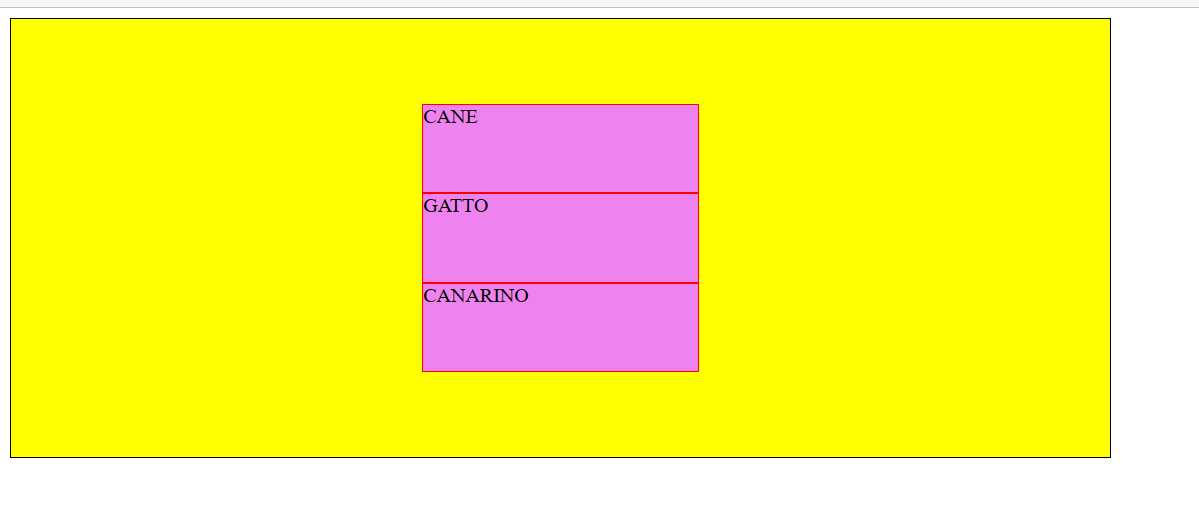


Proviamo alcuni di questi a titolo di esempio.

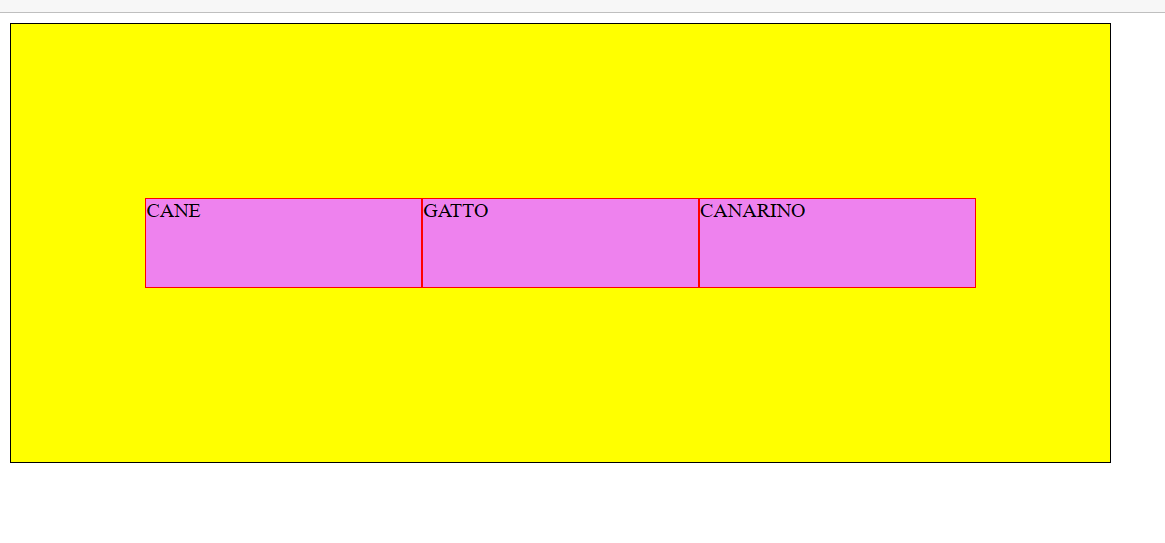
align-items:center:



Combinando con justify-content:center e align-items:center possiamo spostare i nostri div figli al centro del div container:



Combiando flex-direction:row, justify-content:center e align-items:center possiamo ottenere I div in riga e al centro del nostro div:



Ovviamente è impossibile considerare tutte le combinazioni possibili all’interno di un documento word. Quello che conviene fare è provare queste combinazioni nel momento in cui vengono utilizzate in un progetto. Per prima cosa considerare la proprietà flex-direction. Dopodiché si ragiona sulle proprietà justify-content e align-items!

**FLEX WRAP PROPERTY**

Così come flex-direction, anche la proprietà wrap viene inserita al div contenitore.

Il valore di default di wrap è nowrap. Questo significa che, se riduciamo la width del browser, il contenuto non si adatta ad essa automaticamente, ma scompare mano mano che il browser si restringe.

Riprendiamo il nostro esempio valorizzando al valore di default il nostro div parent. Anche in questo caso, la proprietà flex-wrap viene applicata al div container, e ha ripercussione sul comportamento dei div figli. Il valore di default della proprietà wrap è nowrap. Consideriamo quindi il nostro esempio di prima, e aggiungiamo altri due div.

HTML

<body>

    <div id="container">

        <div>CANE</div>

        <div>GATTO</div>

        <div>CANARINO</div>

        <div>CAPIBARA</div>

        <div>ALPACA</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:column;

    justify-content:flex-start;

    align-items:stretch;

    flex-wrap:nowrap;

}

div{

    height:17%;

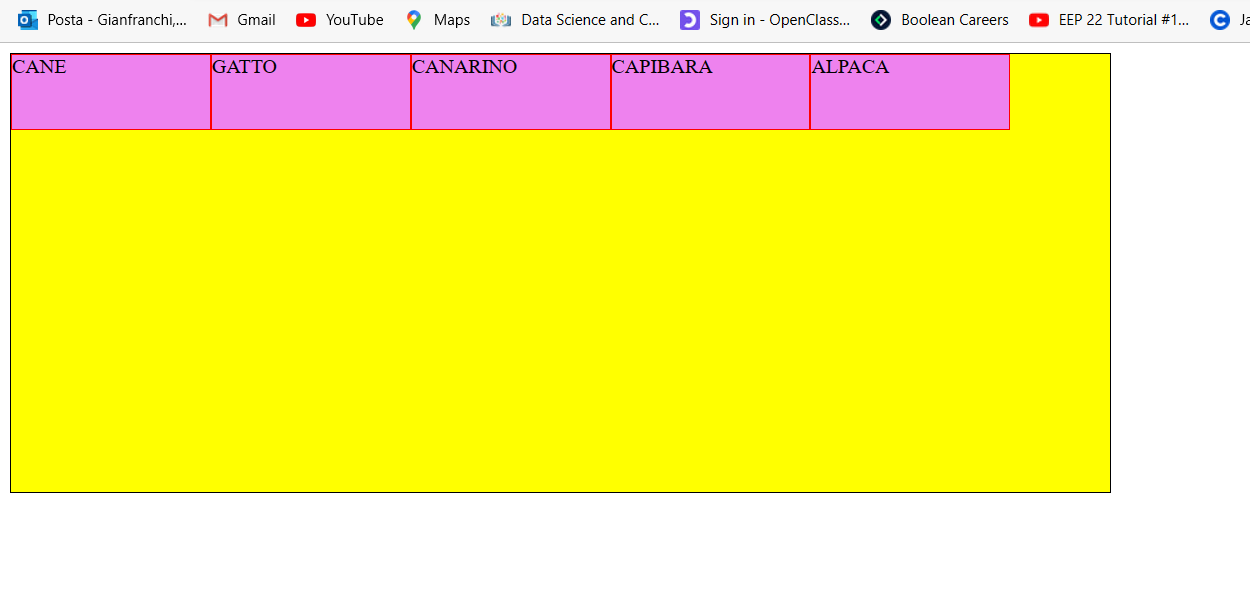
    width:18%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



E questo, ovviamente, equivale ad inserire la proprietà display:flex all’interno del nostro contenitore senza le proprietà flex-direction,juistify-content,align-items oppure wrap.

In questo modo il nostro layout non è responsive. Aumentiamo la width dei div figli:

HTML

<body>

    <div id="container">

        <div>CANE</div>

        <div>GATTO</div>

        <div>CANARINO</div>

        <div>CAPIBARA</div>

        <div>ALPACA</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:row;

    justify-content:flex-start;

    align-items:stretch;

    flex-wrap:nowrap;

}

div{

    height:17%;

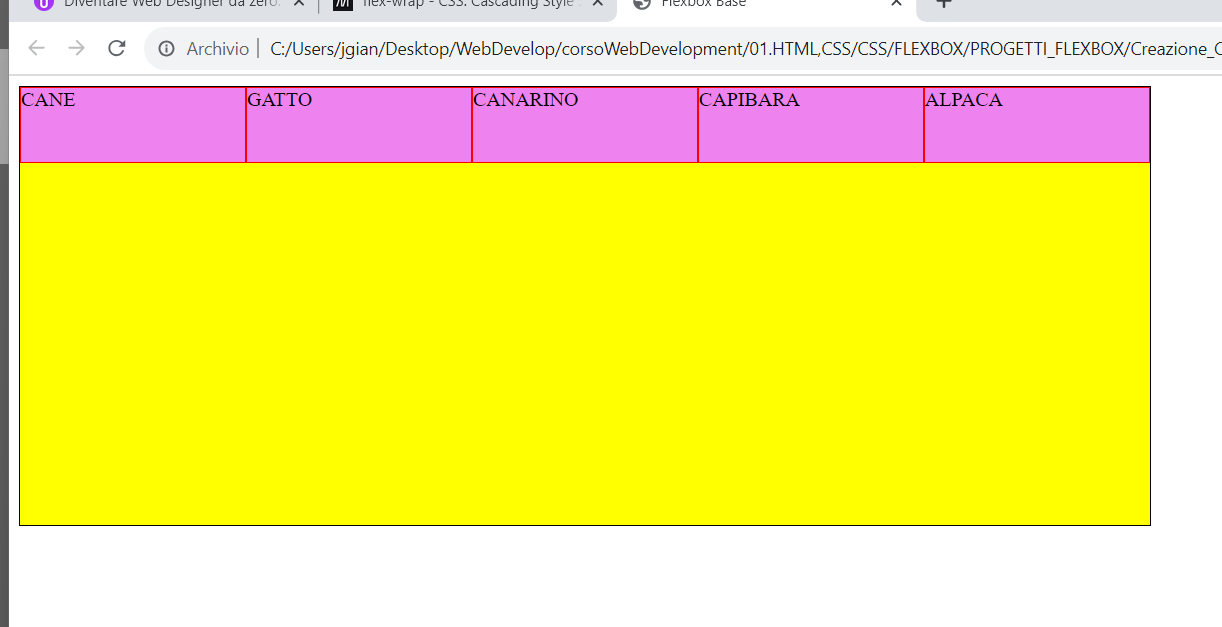
    width:40%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Notiamo che, se la somma delle width dei div figli è superiore alla width del div container, flexbox in automatico fa si che la somma delle width dei div figli sia non superiore alla width del div container, riducendo e rendendo uguali le width dei singoli div container.

Da quanto si evince dal codice css, infatti, la width di ciascun div figlio è pari al 40% della width del div padre, e la width del div padre è pari al 60% della width del browser. Poiché i div figli sono 5, avremo che la width complessiva dei 5 div figli supera di molto la width del div padre.

Per fare in modo che tutti i div figli entrino nel div padre, la width di ciascun div figlio si riduce.

C’è un modo per impedire tutto ciò? Certo! Si utilizza flex-wrap:wrap. In questo modo, la width di ciascun div figlio resta pari al 40% della width del div padre, e gli elementi figli si distribuiranno automaticamente nel div padre mantenendo la loro width originale (40% della width del div padre)

<body>

    <div id="container">

        <div>CANE</div>

        <div>GATTO</div>

        <div>CANARINO</div>

        <div>CAPIBARA</div>

        <div>ALPACA</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:row;

    justify-content:flex-start;

    align-items:stretch;

    flex-wrap:wrap;

}

div{

    height:17%;

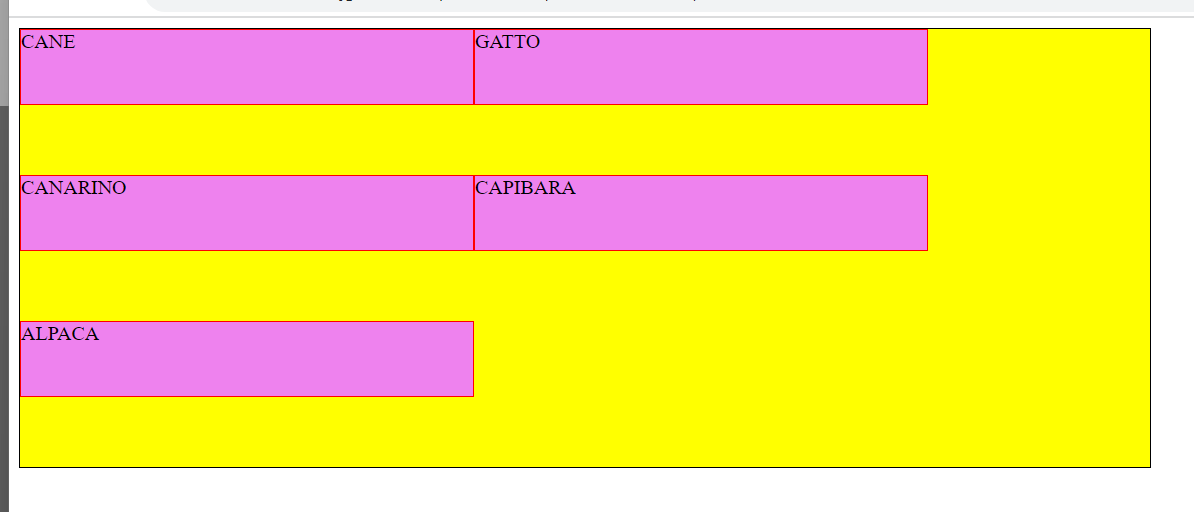
    width:18%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Quindi, i div figli si adattano al div container se è presente la proprietà flex:wrap valorizzata a wrap.

Notiamo ora che è possibile far si che i div figli del nostro div container adattino la loro width nel momento in cui riduciamo la width del browser stesso. Come facciamo? Semplice: Eliminiamo la width dei nostri div figli!!! Aumentiamo anche il numero di div figli per semplicità.

HTML

<body>

    <div id="container">

        <div>CANE</div>

        <div>GATTO</div>

        <div>CANARINO</div>

        <div>CAPIBARA</div>

        <div>ALPACA</div>

        <div>RINOCERONTE</div>

        <div>JENA RIDENS</div>

        <div>GABBIANO</div>

    </div>

</body>

CSS (valori di default delle proprietà)

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-direction:row;

    justify-content:flex-start;

    align-items:stretch;

    flex-wrap:wrap;

}

div{

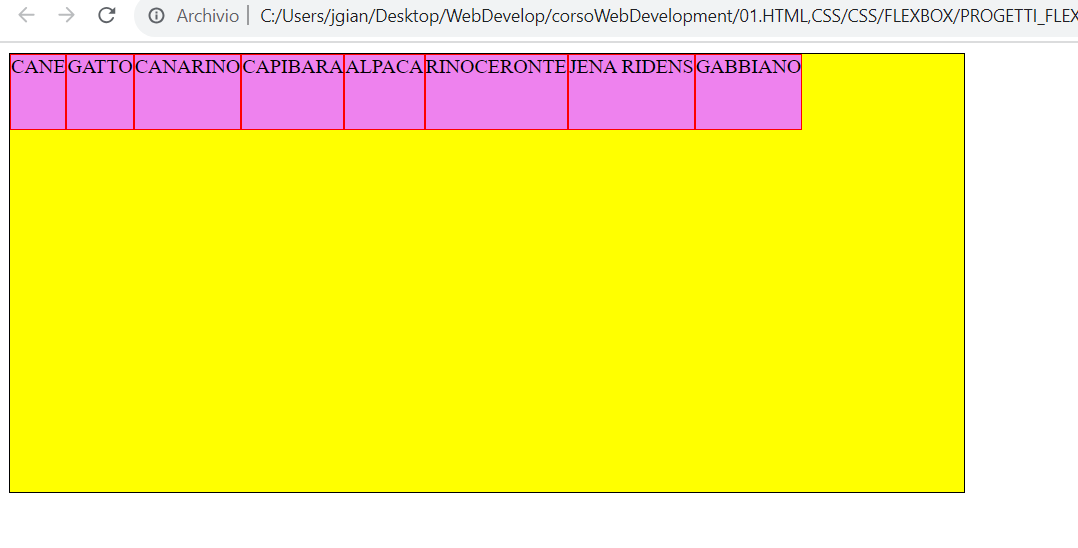
    height:17%;

    border:1px solid red;

    background-color:violet;

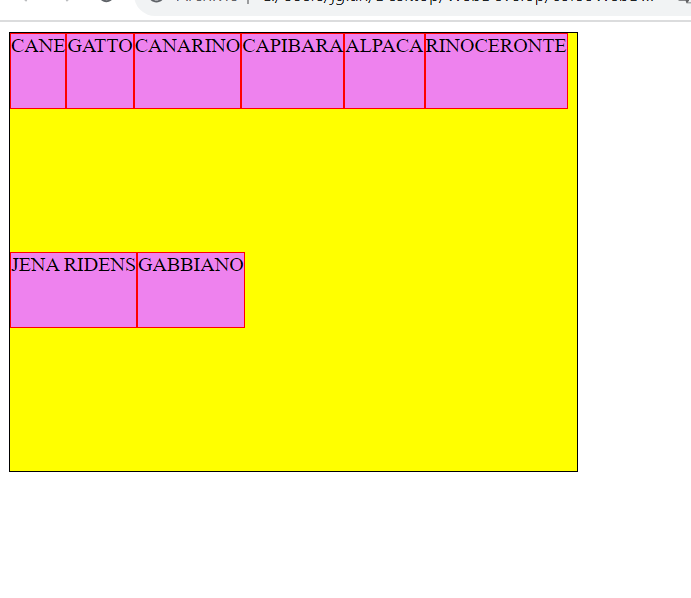
}

BROWSER



Ricordiamo che, se la width dei div figli non viene specificata, essa si adatta al loro contenuto.

Proviamo ora a ridurre la width del browser



Perfect! Quando il browser tocca uno dei div figli, quest’ultimo va giu.

Esiste anche la proprietà wrap-reverse. Provarla per capire come funziona!

Si noti ora che le proprietà flex-direction e flex-wrap possono essere inserite su una sola riga.

Invece di specificarle tutte e due, è possibile utilizzare la proprietà flex-flow. Questa proprietà accetta due valori: il primo riguarda il flex-direction, il secondo riguarda il flex-wrap. Di default, il nostro CSS può essere quindi specificato nel seguente modo:

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:column nowrap;

    justify-content:flex-start;

    align-items:stretch;

}

Applicandola al nostro esempio:

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:row wrap;

    justify-content:flex-start;

    align-items:stretch;

}

div{

    height:17%;

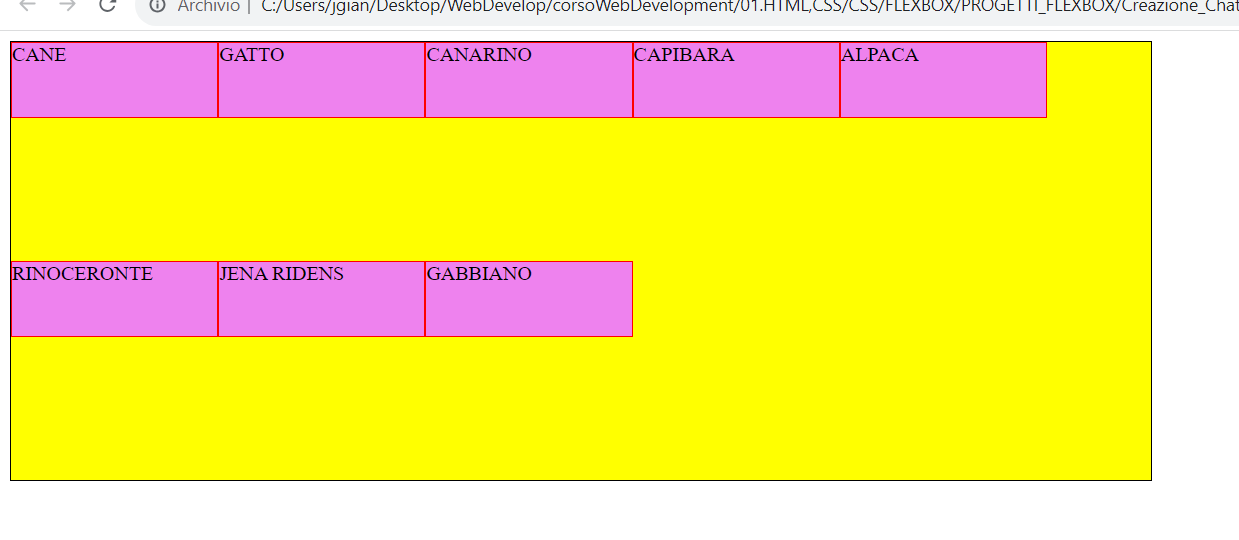
    width:18%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Ricordarsi di provare wrap-reverse

**ALIGN CONTENT PROPERTY**

Align content si utilizza in sinergia con flex-wrap. Esso può assumere tre valori: stretch, flex-start e flex-end. Il valore di default è ovviamente stretch. Il nostro codice con i valori di default del nostro div parent è quindi il seguente:

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:column norap;

    justify-content:flex-start;

    align-items:stretch;

    align-content:stretch;

}

Proviamo quindi ad utilizzare il nostro align-content nel caso più semplice, ossia in cui abbiamo flex-flow:row wrap.

Nel primo caso avremo align-content: flex-start, nel secondo align-content:flex-end, nel terzo align-content:center, nel quarto caso align-content:space-around e nell’ultimo space-between

align-content:flex-start

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:row wrap;

    justify-content:flex-start;

    align-items:stretch;

    align-content:flex-start;

}

div{

    height:17%;

    width:20%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



align-content:flex-end

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:row wrap;

    justify-content:flex-start;

    align-items:stretch;

    align-content:flex-end;

}

div{

    height:17%;

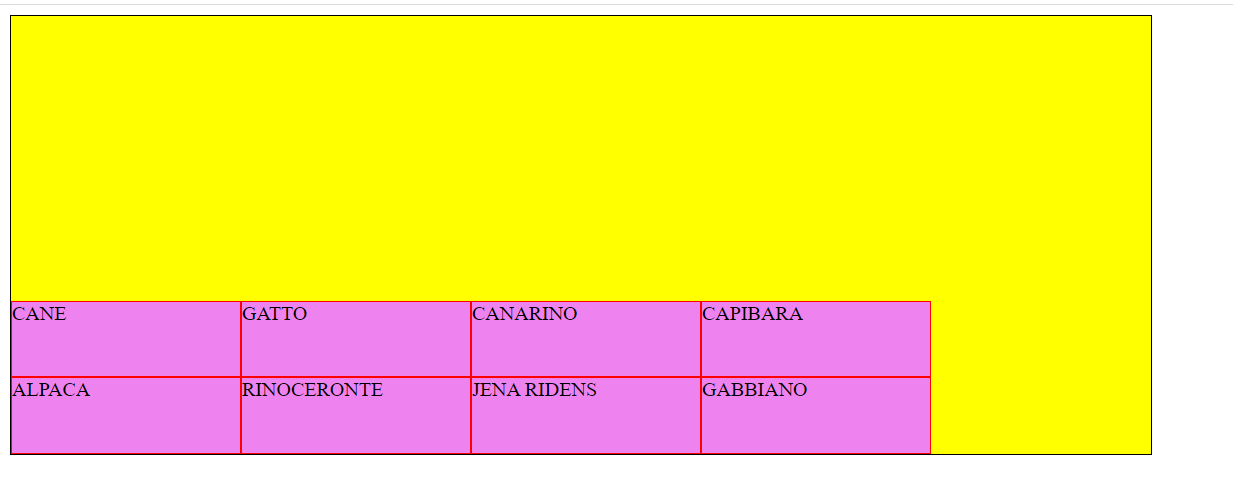
    width:20%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



align-content:center

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:row wrap;

    justify-content:flex-start;

    align-items:stretch;

    align-content:center;

}

div{

    height:17%;

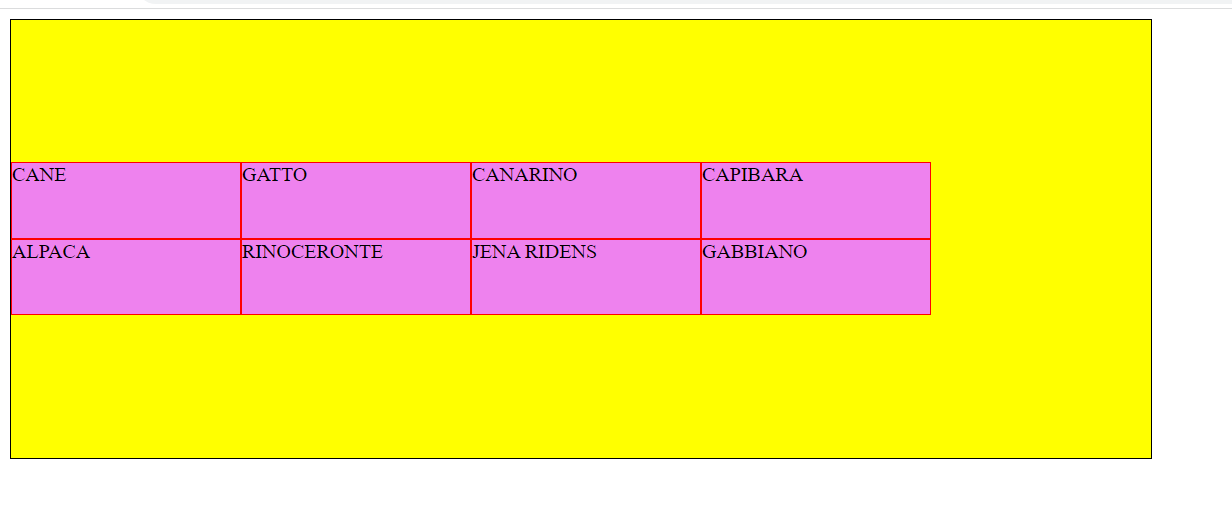
    width:20%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



align-content:space-around

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:row wrap;

    justify-content:flex-start;

    align-items:stretch;

    align-content:space-around;

}

div{

    height:17%;

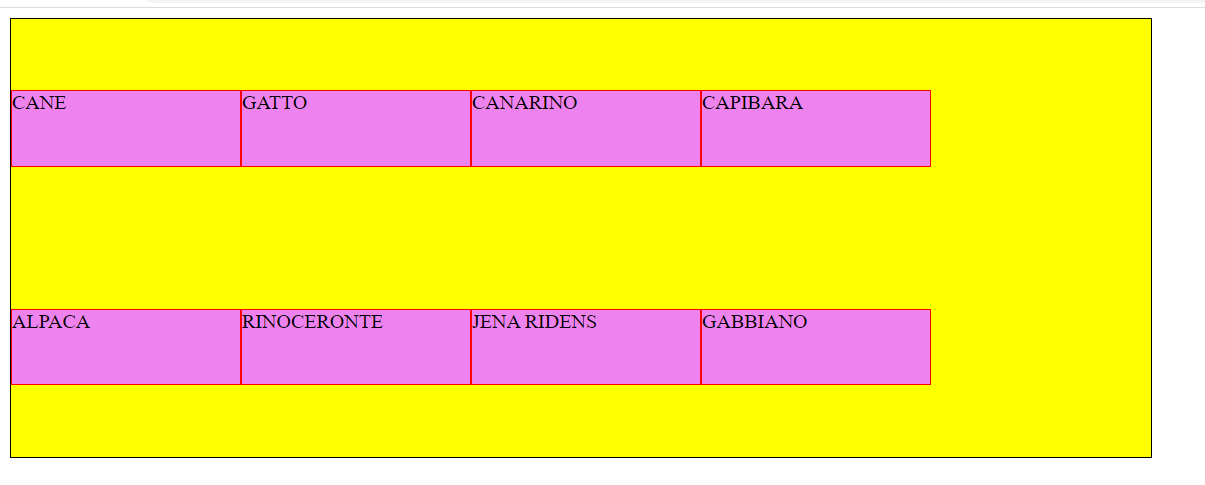
    width:20%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



Align-content:space-between

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:row wrap;

    justify-content:flex-start;

    align-items:stretch;

    align-content:space-between;

}

div{

    height:17%;

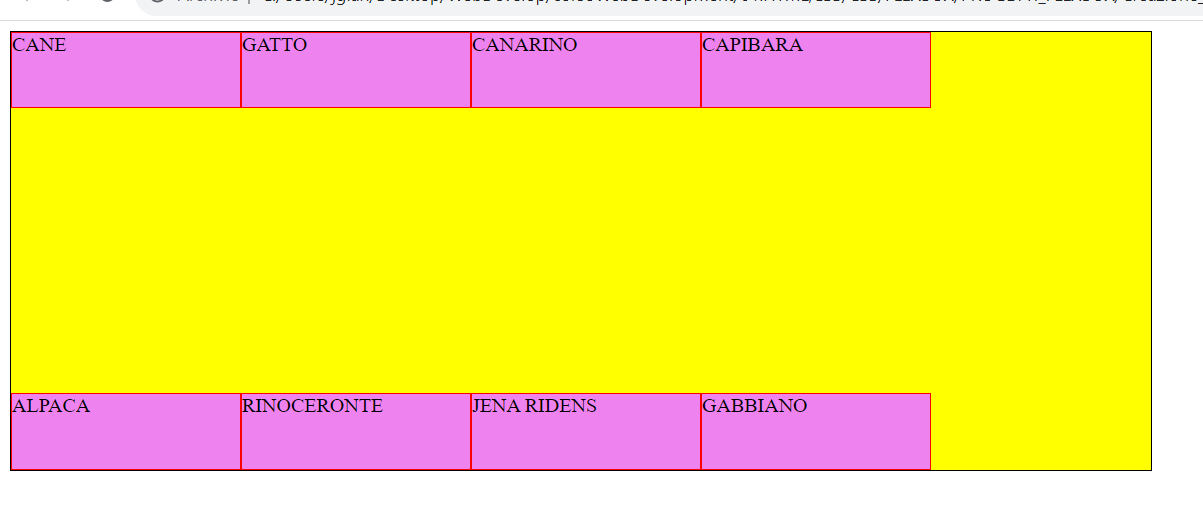
    width:20%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



**PROPRIETA’ FLEXBOX APPLICABILI AI DIV CONTENUTI NEL CONTAINER**

Torniamo al nostro esempio iniziale, con alcune modifiche riguardanti il nostro codice HTML. Come al solito, il nostro container avrà le proprietà flexbox valorizzate al loro valore di default.

HTML

<body>

    <div id="container">

        <div class="figlio cane">CANE</div>

        <div class="figlio gatto">GATTO</div>

        <div class="figlio canarino">CANARINO</div>

        <div class="figlio capibara">CAPIBARA</div>

        <div class="figlio alpaca">ALPACA</div>

        <div class="figlio rinoceronte">RINOCERONTE</div>

        <div class="figlio jena-ridens">JENA RIDENS</div>

        <div class="figlio gabbiano">GABBIANO</div>

    </div>

</body>

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:column nowrap;

    justify-content:flex-start;

    align-items:stretch;

    align-content:stretch;

}

.figlio{

    height:17%;

    width:20%;

    border:1px solid red;

    background-color:violet;

}

BROWSER



**ORDER**

La proprietà order serve a dare importanza a i div figli contenuti nel div container. Di default, ciascun elemento figlio ha la proprietà order valorizzata a 0. Possiamo applicare questa proprietà ai nostri div figli per dargli un ordinamento a nostro piacimento.

Nel nostro esempio, i div figli sono applicati nel div container hanno il seguente ordine.

1. CANE
2. GATTO
3. CANARINO
4. CAPIBARA
5. ALPACA
6. RINOCERONTE
7. JENA RIDENS
8. GABBIANO

Supponiamo di voler cambiare l’ordine dei div figli. Il nuovo ordine sarà il seguente:

1. JENA RIDENS
2. CANE
3. GABBIANO
4. GATTO
5. CANARINO
6. RINOCERONTE
7. CAPIBARA
8. ALPACA

Per farlo, si utilizza la proprietà order sui div figli:

CSS

#container{

    border:1px solid black;

    width:60%;

    height:350px;

    background-color:yellow;

    display:flex;

    flex-flow:column nowrap;

    justify-content:flex-start;

    align-items:stretch;

    align-content:stretch;

}

.figlio{

    height:17%;

    width:20%;

    border:1px solid red;

    background-color:violet;

}

.cane{

    order:2;

}

.gatto{

    order:4;

}

.canarino{

    order:5;

}

.capibara{

    order:7;

}

.rinoceronte{

    order:6;

}

.alpaca{

    order:8;

}

.jena-ridens{

    order:1;

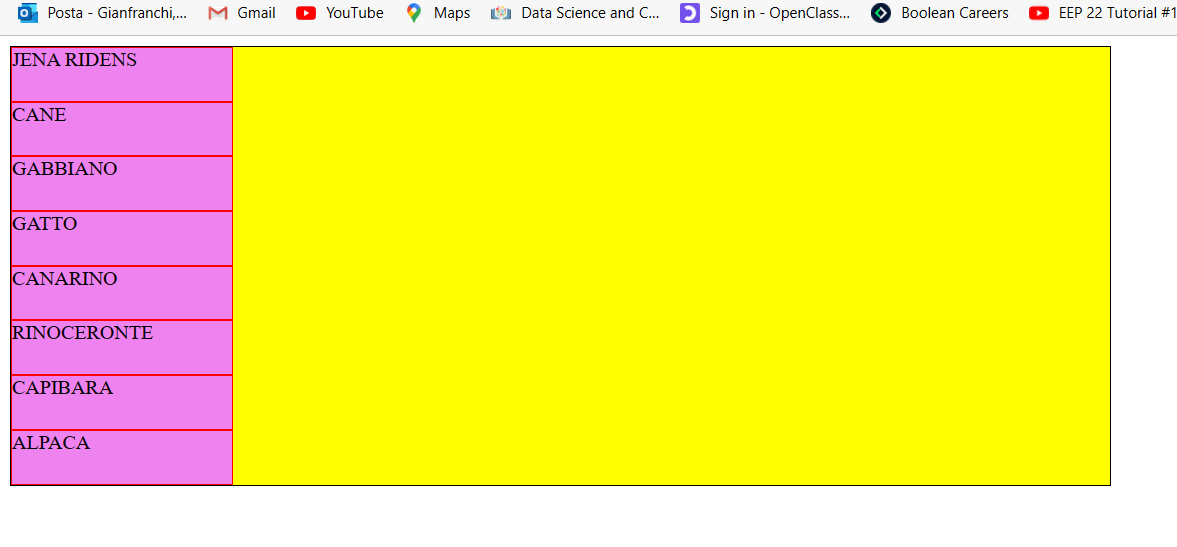
}

.gabbiano{

    order:3;

}

BROWSER



CVD