flexbox

→ It deals with layout in one dimension at a time - either as a row or a column.

```
<div class = " flex - container ">

        <div>1 <div>
        <div> 2 <div>
        <div>3 <div>
<div>
```

→ first you need to define a flex - container

⇓

so, this is a flex container with 3 flex - items

```
.flex - container {

        display: flex or inline - flex
}
```

flex → It will take complete available width like block elements

inline - flex → It will take only the required width like inline elements

Normal

```
<div>
    <div> item1 </div>
    ____________ 2 ______
    ____________ 3 ______
</div>
```

⇒

```
item 1
item2
item3
```

Style
Date:..................
Page No:..................

display : flex

```
<div class="flex-container">
    <div> item1 </div>
    <div> item2 </div>
    <div> item3 </div>
</div>
```

```
→ .flex-container {
        display : flex;
  }
```

o/p →

item1 item2 item3

takes complete available width

```
→ .flex-container {
      display : inline-flex;
  }
```

o/p :-

item1 item2 item3

Takes only required width

• **Properties related to**

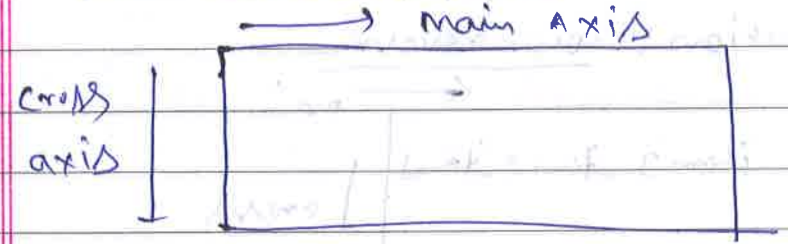
flex-container → flex-direction
flex-wrap
~~ali~~ align-items
justify-content

flex-item → order
align-self

① **flex-direction**

row (default)
row-reverse
column
column-reverse

* flex-direction : row



```
<div class="flex-container">
  <div> item1 </div>
  ———— 2 ————
  ———— 3 ————
</div>
```

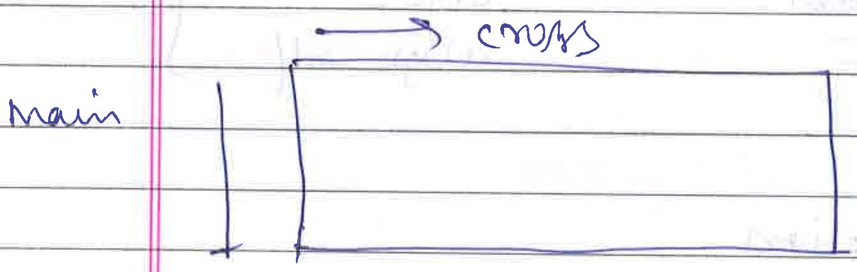
⟹ | item1 item2 item3 |

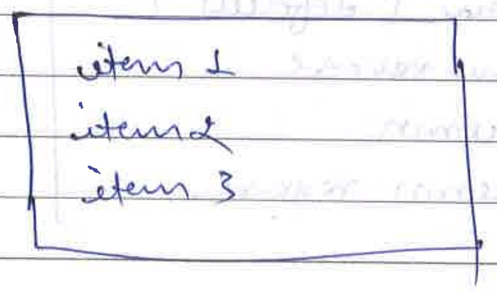
```
.flex-container {
    display : flex
    flex-direction : row (default)
}
```

Brilliant

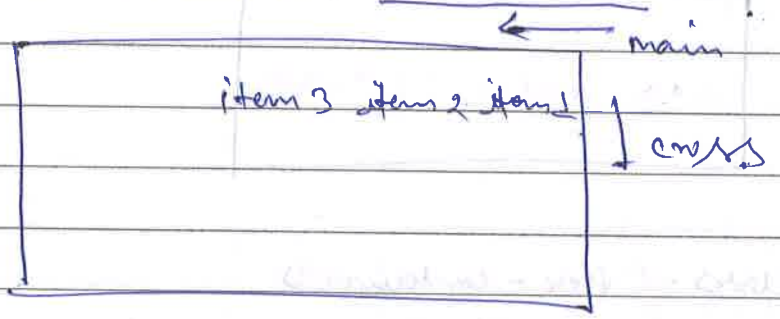
* flex-direction : column

```
• flex-container {
        display : flex
        flex-direction : column;
  }
```

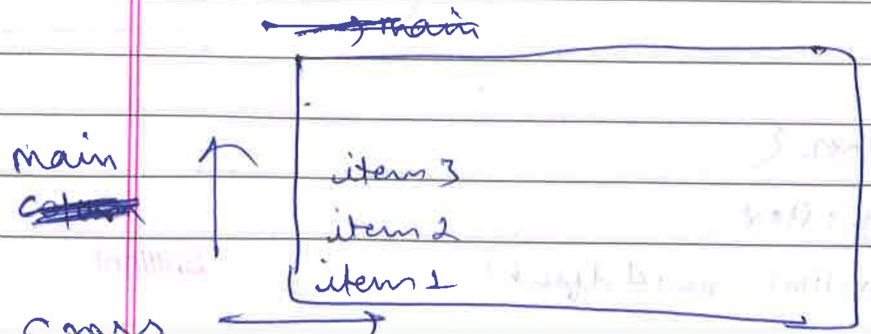




* flex-direction : row-reverse



* flex-direction :- column-reverse



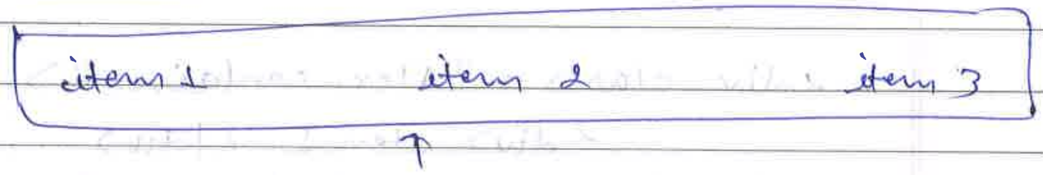
(2) flex-wrap:- default value : no

$$\begin{tikzcd} & \text{no} \\ \text{flex-wrap} \arrow[ur, dash] \arrow[dr, dash] & \\ & \text{wrap} \end{tikzcd}$$

```
<div class =" flex-container ">
   <div style =" width : 300px "> item 1 </div>
   ————————————————————————— 2 ———————
   ————————————————————————— 3 ———————
</div>
```

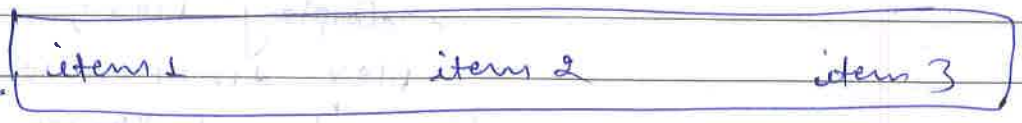
```
•flex - container {
        display : flex
        flex-direction : row    ← default
        flex- wrap : wrap ;
  }
```

with no-wrap

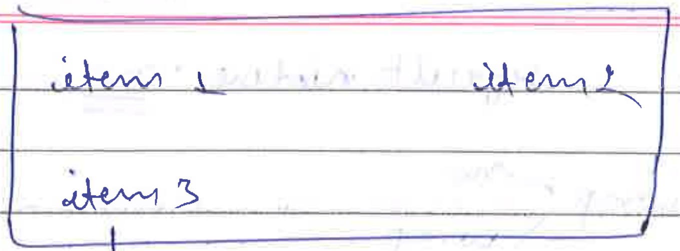


It will be like this only even on decreasing viewport size

with wrap:-



on small size screen, it will wrap such that it will occupy 300px.



This has been wrapped because it need 300 px width.

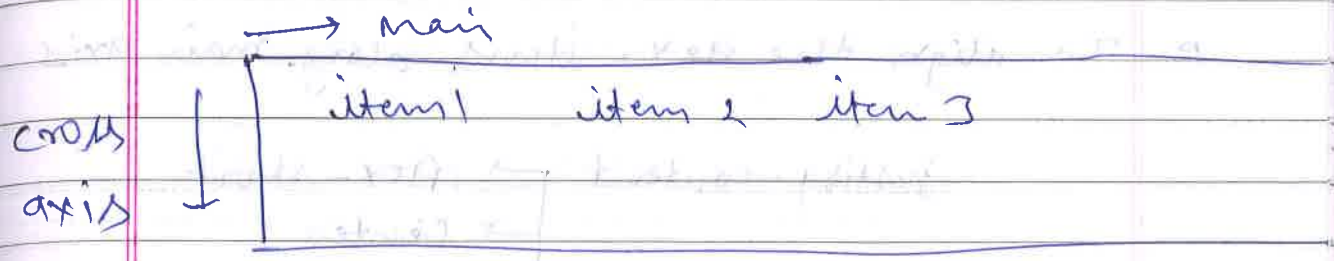
③ **align-items** :- To ~~assign~~ align all the flex items along cross axis

~~stretch~~
- stretch → (default)
- flex-start → at the start of cross-axis
- flex-end → at the end of cross axis
- center → at the center of cross axis

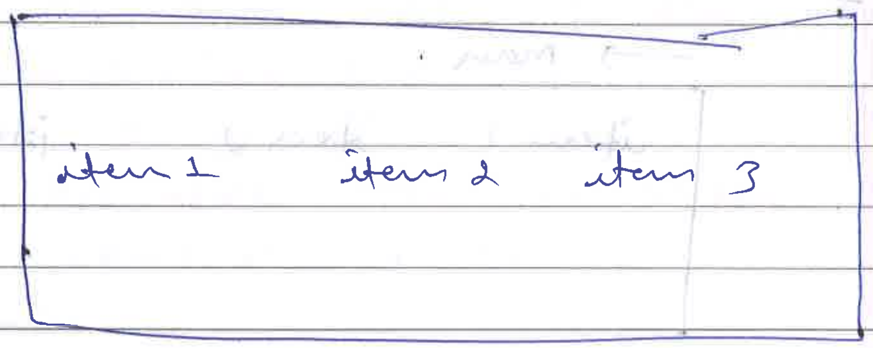
```
<div class = "flex-container">
      <div> item1 </div>
      ________ 2 ________
      ________ 3 ________
</div>

.flex-container {
            display: flex;
            flex-direction: row-default
      align-items: stretch (default)
                   flex-start
                   flex-end
                   center
      width: 200px
}
```

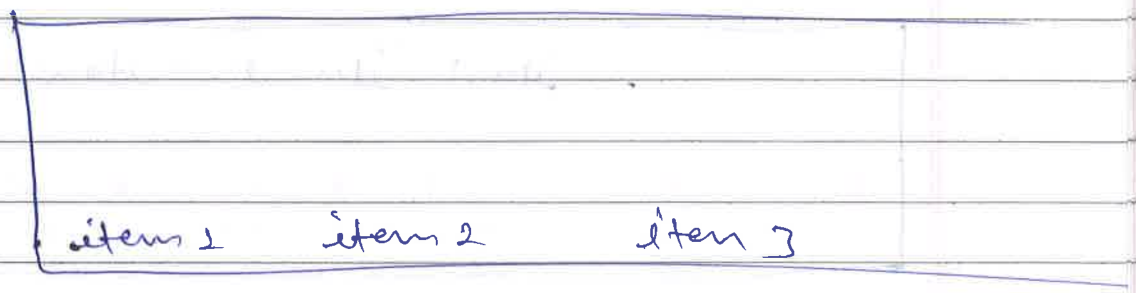
→ align-items : flex-start



→ align-items : center



→ align-items : flex-end

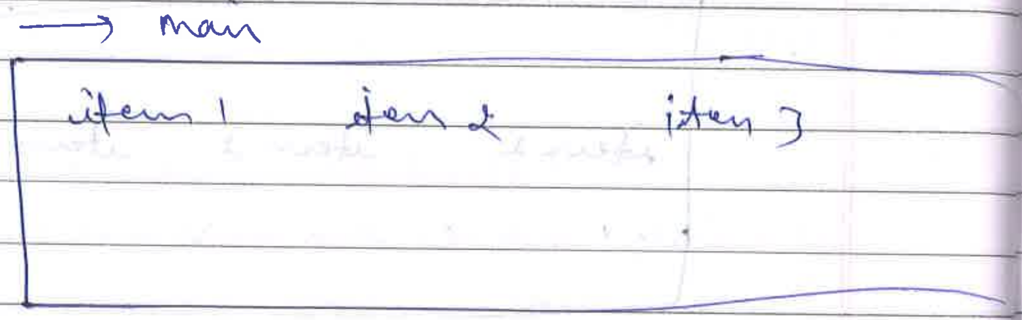


④ justify-content :-

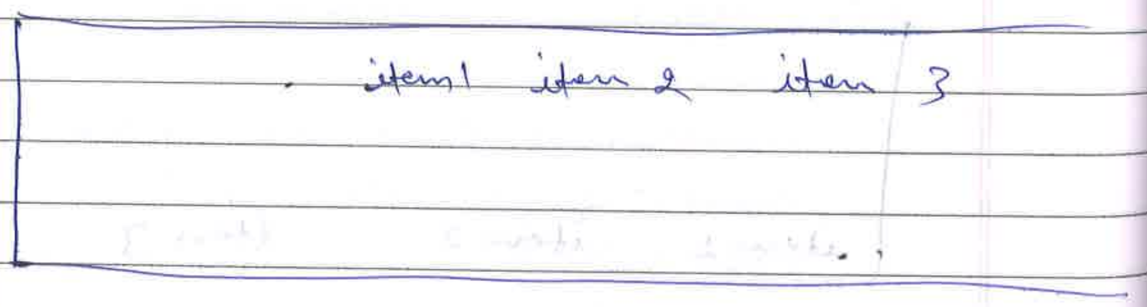
• To align the flex-items along main axis

justify-content
- → flex-start
- → center
- → flex-end

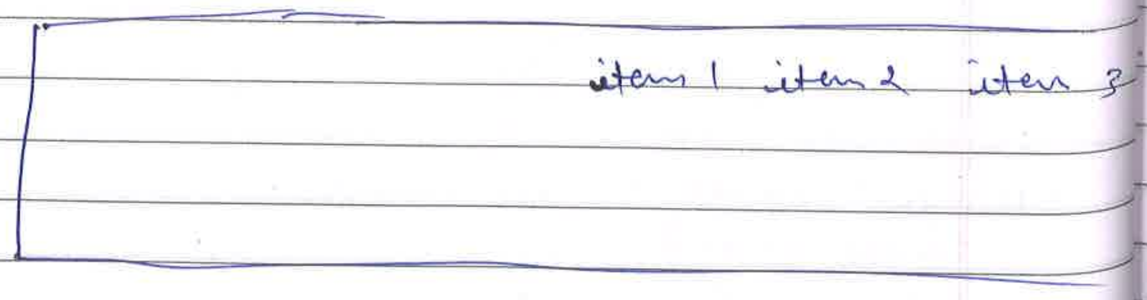
flex-start



center



flex-end



* properties of flex-items :-

① order → default value is 0

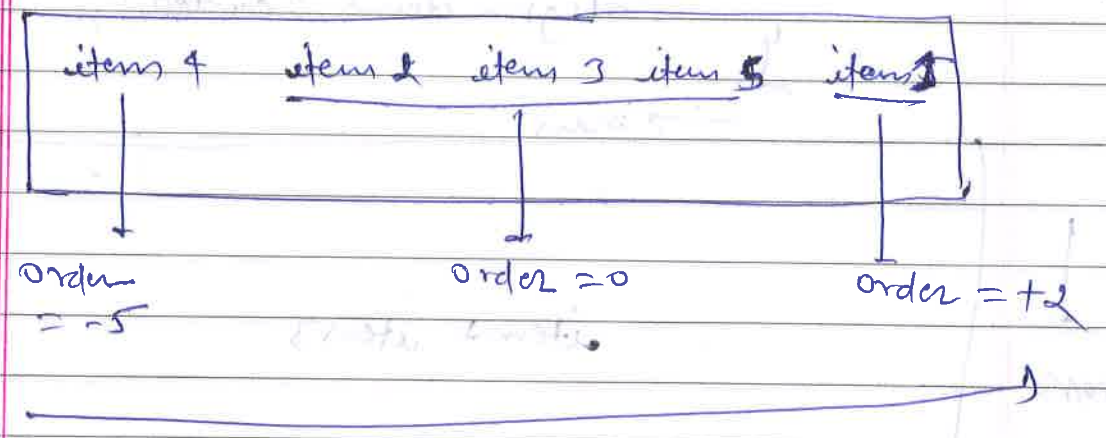
② align-self

① order

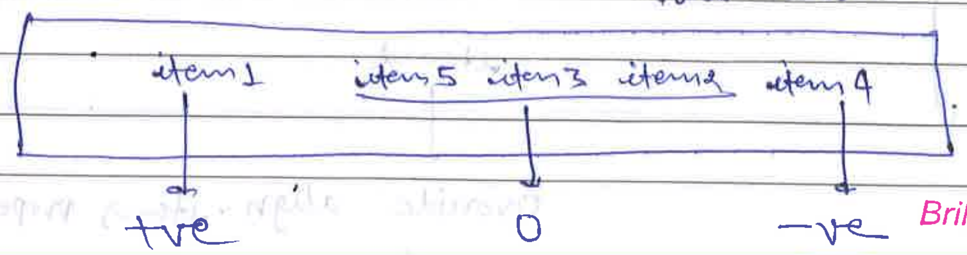
```
<div class = "flex-container">
   <div style = "order : 2"> Item 1 </div>
   <div> Item 2 </div>
                    3
    <div style = "order : -5"> Item 4 </div>
     <div> item 5 </div>

      .flex-container {
           display : flex;
            display-direction : ;
```

row



row-reverse



→ flex-start
→ flex-end
→ center

② **align-self :-**

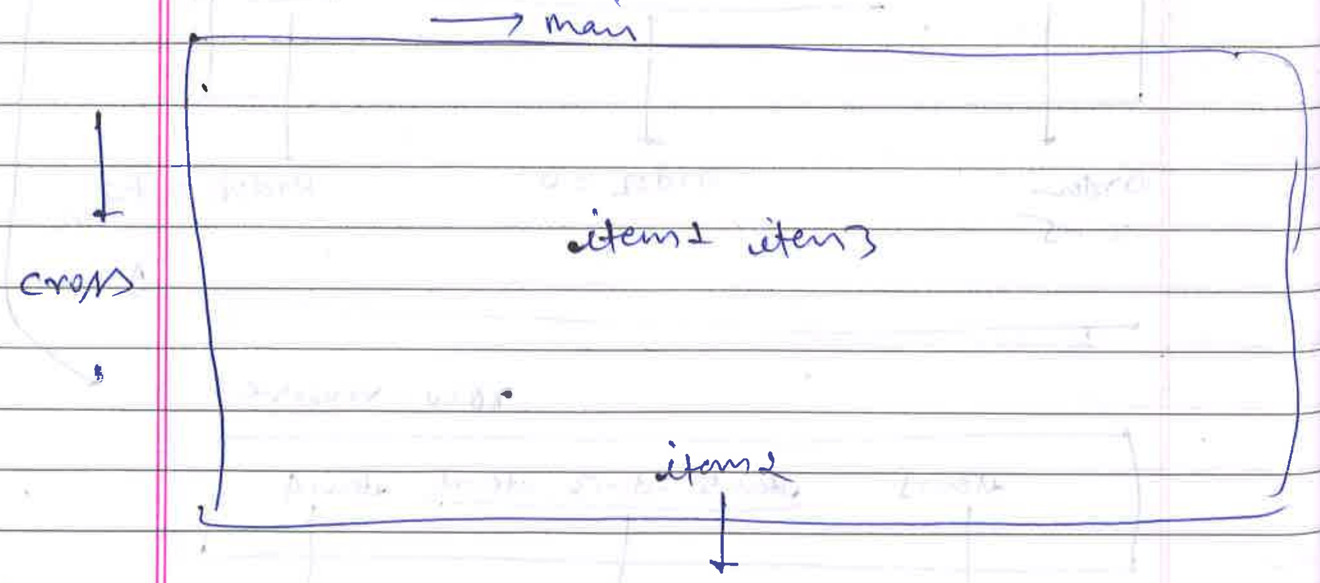
* used to align any flex-item along cross axis

* It will override align-items property if it is there.

```
<div class="flex-container">
    <div> Item1 </div>
    <div style="align-self : flex-end"> Item2 </div>

    <div> Item3 </div>
</div>

.flex-container {
    height : 500 px;
    display : flex;
    flex-direction : row;
    align-items : center;
}
```



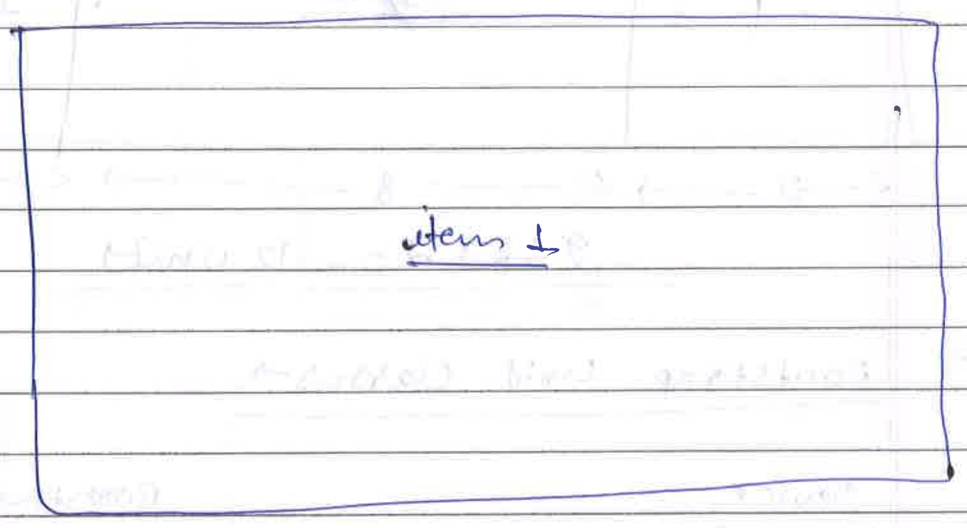
Override align-items property of flex-contai

→ **One Imp interview question:-**

How to perfectly center an item

```
<div class="flex-container">
    <div> Item </div>
<div>

.flex-container {
    height : 500 px;
    display : flex;
    [ align-items : center;
      justify-content : center ] ⇒ center
}
```



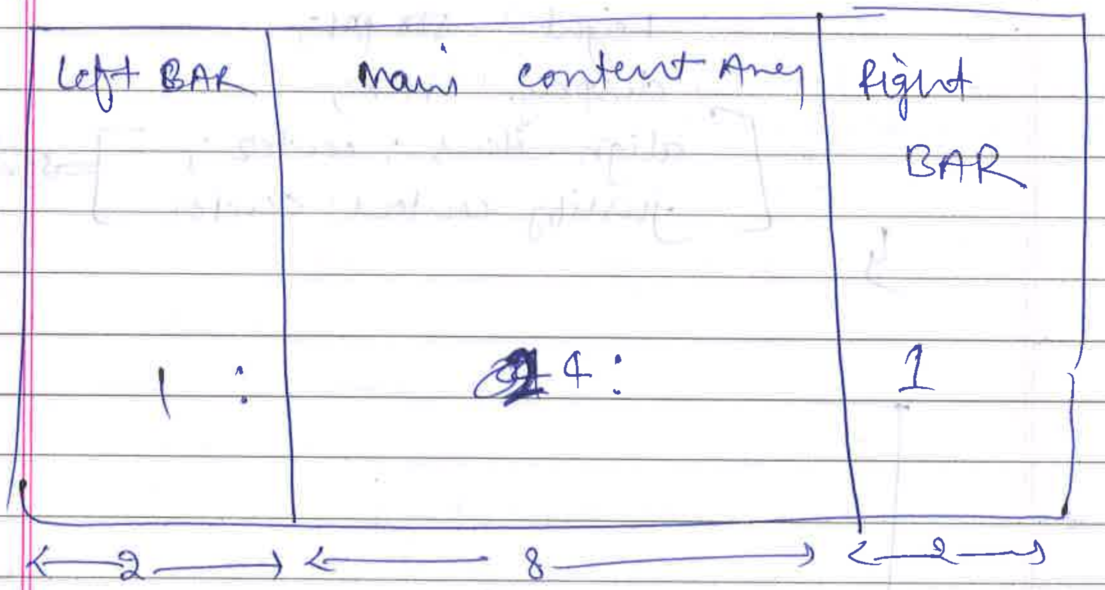
~~Media query~~.

Bootstrap – Grid-system.

→ used for creating page layout through a series of rows and columns.

→ This grid ~~column~~ system consist of 12 columns.

Ex. 3-column layout → using this grid system.



2 + 8 + 2 = 12 units

→ Bootstrap Grid classes →

Device	Bootstrap class
Extra small devices (<768 px) —mobiles	→ .col-xs-*
Small devices (≥768px) tablets	→ .col-sm-*
medium devices – Desktop (≥992 px)	→ .col-md-*
Large device (≥1200px) Desktop	→ .col-lg-*

→ * → मतलब है कितने columns चाहिए.
जैसे example में हम एक row में तीन column चाहिये थे 1

```
<div class =" Container">
    <div class = "row">
        <div class = " col-md-2 "> Left side Bar </div>
        <________________________8 – main content </div>
        _________________________2 – Right ___________
    </div>
</div>
```

* अगर एक row में हम 13 use करते है instead of 12 तो last wala column new line में wrap हो जायेगा

Ex. 2