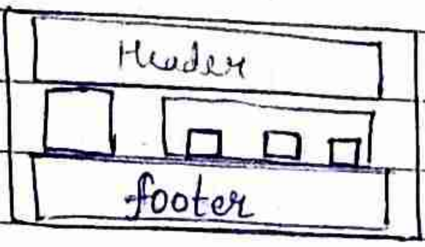
Grid in CSS

Flex-box = 1D layout design tool
grid = 2D layout design tool.



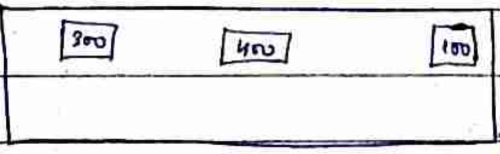
→ Grid example

Grid बनाने के लिए container को display: grid; करना पड़ता है ॥

Grid Container properties

1) grid-template-columns : $\overset{C_1}{300px}$ $\overset{C_2}{auto}$ $\overset{C_3}{100px}$;

→

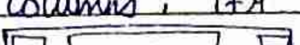


→ 800px

3 columns बनेगा
Ist → takes 300px
2nd → takes auto
3rd → takes 100px

or

grid-template-columns : 1fr 4fr 1fr ;

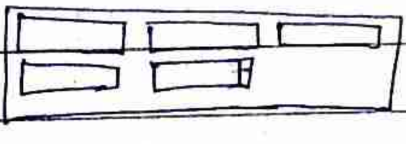


पूरा width को 1:4:1 में बाँट दिया

2) grid-template-columns : repeat($\overset{\text{no of columns in 1 row}}{3}$, auto); → ~~size~~ width of column

3 - 3 columns बनाएगा जिसका size बराबर होगा

If there are 5 html elements (i.e div)



V41

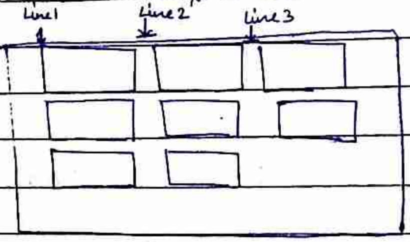
3. grid-gap : 2rem; // gives horizontal & vertical gap of 2rem to all grid-items.
grid-column-gap : क;
grid-row-gap : ख;

4. grid-template-rows : 1fr 1fr 4fr;
⇒ 3 rows ~~बना~~, height 1:1:4 ~~है~~ के बनेंगे

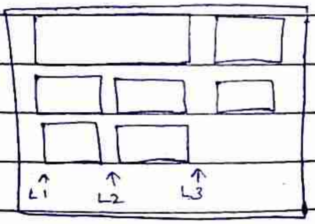
5. grid-auto-rows : 2fr;
⇒ बाकी जितने भी rows होंगे उन्हें 2:2:2.. ~~में~~ के height में बाँट देगा

V42

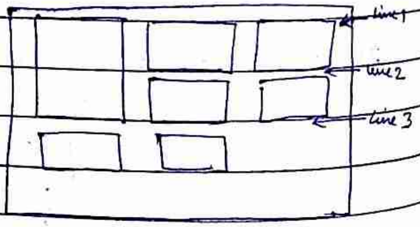
Spanning in Grid



```
.box:first-child{
    grid-column-start : 1;
    grid-column-end : 3;
}
```



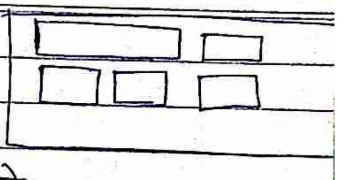
```
.box:first-child{
    grid-row-start : 1;
    grid-row-end : 3;
}
```



Better Way ⟹

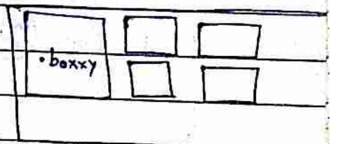
```
.box:first-child{
} grid-column: 1/span 2;
```

Column 1 से start हो रहा है — 2 columns को merge/span करके

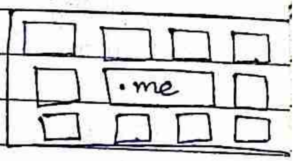


```
.boxxy{
    grid-row: 1/span 2;
}
```

row 1 से start हो रहा है — 2 rows को खा के



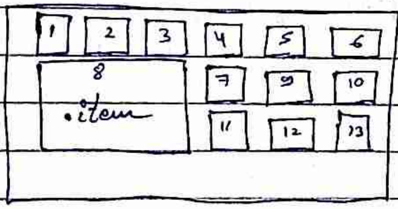
```
.me{
    grid-column: 2/span 2;
    grid-row: 2;
}
```



Grid Area (दोनों को मिला दिया)

grid-area : starting row / starting column / row span / column span;

```
Eg. .item{
    grid-area: 2/1/span 2/span 3;
}
```



JH ⇒ Justify से horizontal

Date :
Page :

For aligning (vertical/horizontal) of ~~either grid items~~ ~~or grid~~ we use -

~~justify-content: align-items~~

refer Page. 24

Horizontal
→ justify-content (works with multi line content)
→ justify-items

Vertical
→ align-content (works with multi-line content)
→ align-items

justify-items · align-items · start/end/Center/stretch.

justify-content · align-content · center/start/end/space-evenly/space-around/space-between

For a Particular item

This property applies to a grid item inside a single cell.

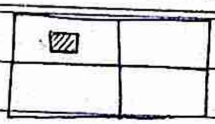
place-self : (start/end/center/stretch) (start/end/centre/stretch);
vertical alignment — Horizontal alignment

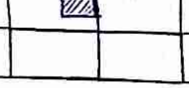
or

place-self : start/end/center/stretch ;
both (~~to~~ vertical as well as horizontal in 1 value).

vvi Respousiveness

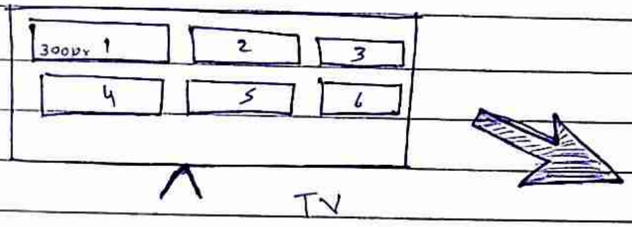
Date :
Page : 32

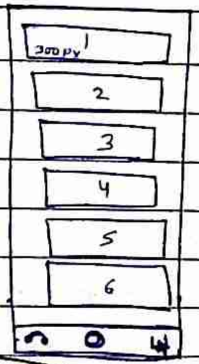
Eg.
```
.item-a{
    place-self : center center;
}
```
OR
```
item-a{
    place-self : center ;
}
```


Eg-
```
.item-a{
    place-self : centre end ;
}
```


VH3

Auto-fit & minmax() in Grid.

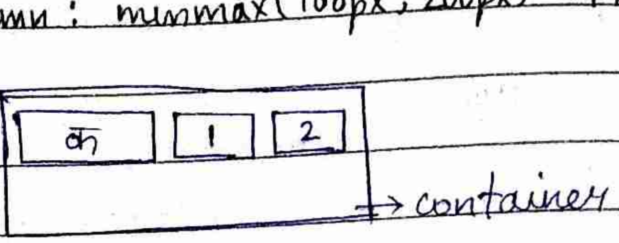




vvi
```
.container{
grid-template-columns :
        repeat(auto-fit, minmax(300px, 1fr));
}
```
mobile

जैसे / ही screen width size 300px से कम हुआ,

```
.Container {
    grid-template-column: minmax(100px, 200px) 1fr 1fr;
}
```



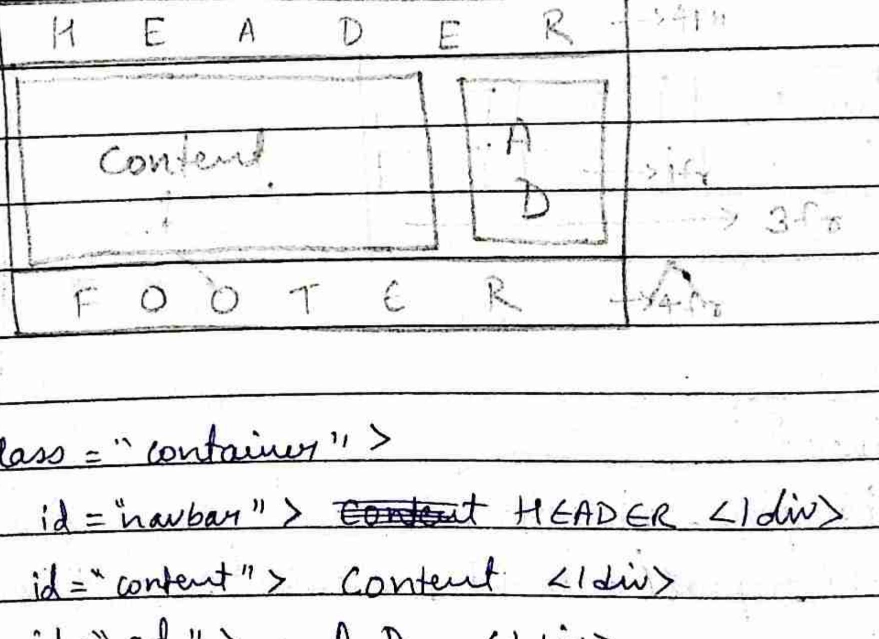
जब container को resize किया जाएगा, तब $\boxed{1}$ $\boxed{2}$ container के width के हिसाब से छोटा-बड़ा होगा (width) लेकिन,

जो $\boxed{क}$ है, ~~क~~ उसका width minimum 100px और maximum 200px ही ~~होगा~~ रहेगा चाहे container का width कितना भी छोटा या बड़ा हो जाए

100px - 200px के बीच ही resize होगा $\boxed{क}$

√44

Grid-template-areas



```
<div class="container">
    <div id="navbar"> ~~Content~~ HEADER </div>
    <div id="content"> Content </div>
    <div id="ad"> A D </div>
    <div id="footer"> FOOTER </div>
</div>

~~.container { display: grid; }~~
#navbar { grid-area: navbar; }
#content { grid-area: ~~navbar~~ content; }
#ad { grid-area: ad; }
#footer { grid-area: footer; }
```

Date :
Page :

```
• container {
      display : grid;
      grid-gap : 1rem;
      grid-template-areas :
            ` navbar navbar navbar navbar'
            ` content content content ad'
            ` footer footer footer footer' ;
}
```