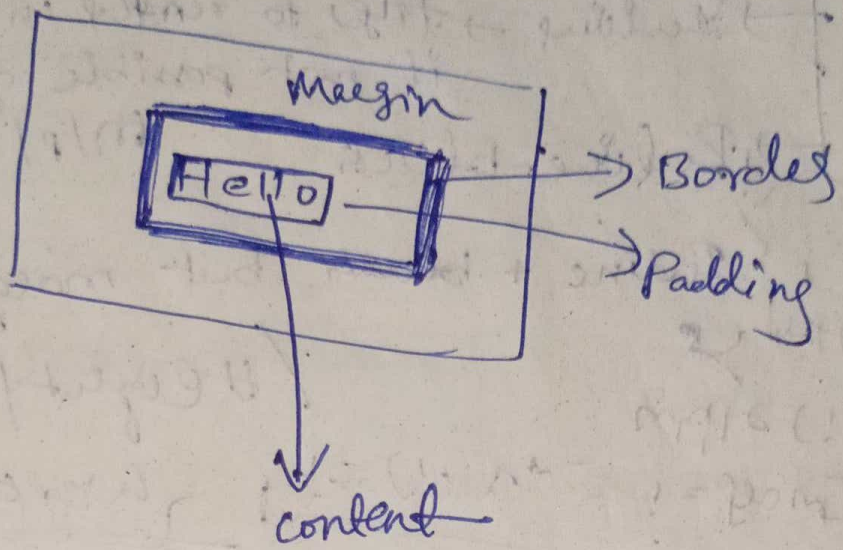
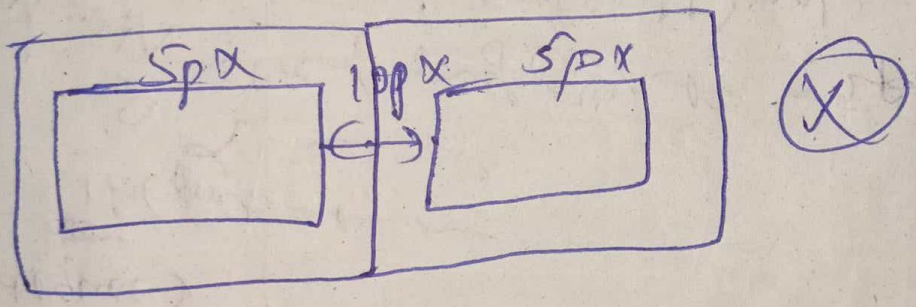
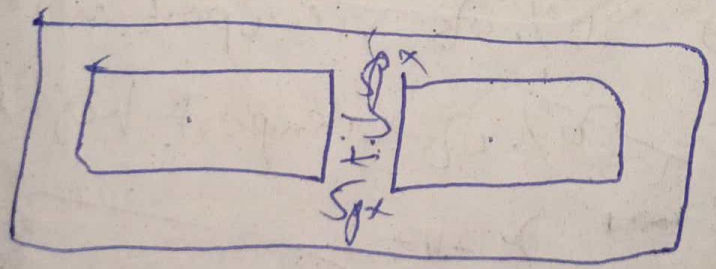


In span tag height & width doesn't work





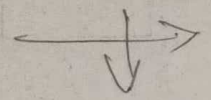
In span tag we can't use height & width but using display inline and display block property we can use height and width inside span tag.

* no wrap packs boxes squeeze and won't let go to new line.

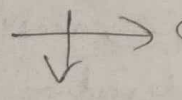
flex-wrap: wrap-reverse

---

flex-direction: row then



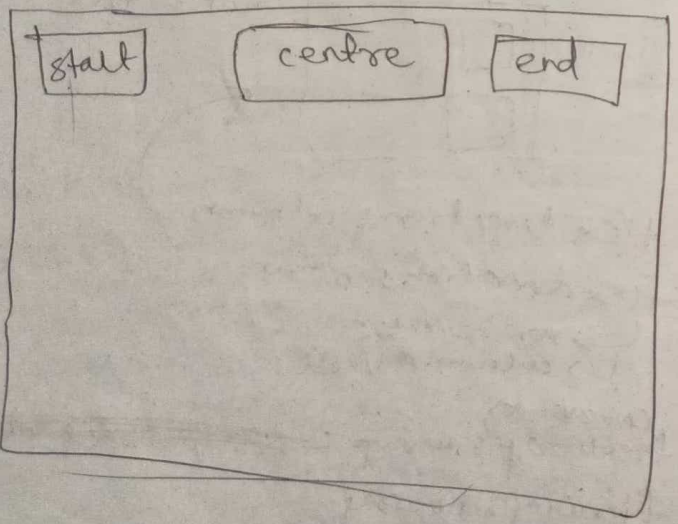
flex-direction: column then



---

consides flex-direction as row

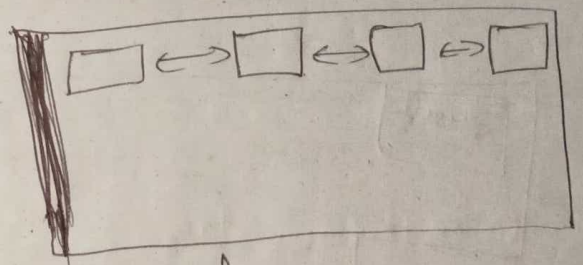
* Justify-content:



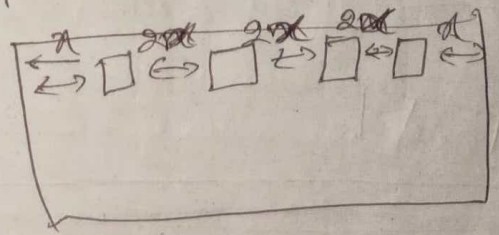
when flex direction is row then main axis is horizontal axis and justify-content works along main axis, if flex-direction is column main axis is vertical and justify content works along main axis.

justify content works with main axis

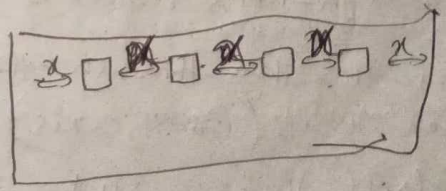
* justify-content: space-between



* space-around



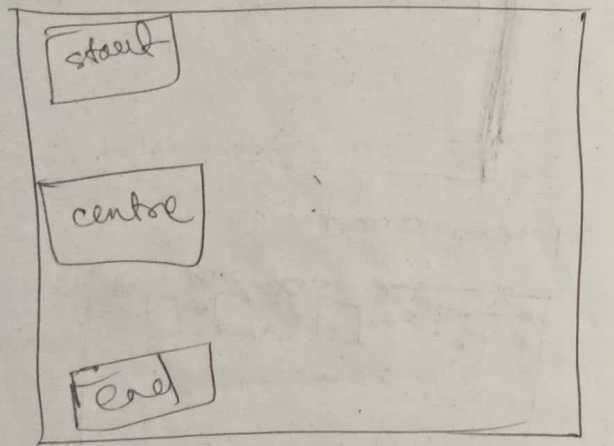
* space-evenly



* align-items : flex start
flex end
centre

~~align items~~ when flex is in row direction then align items works along y axis.

align items works with cross axis



* ~~stretch~~ : align-items : stretch

stretches flex items through the axis in which it is (cross axis)

* by default items will be set to stretch

* flex items properties

→ order : numbers
more orders, more last in line that flex item will be there

→ flex ~~grow~~ shrink : numbers
shrinks or grows/expands numbers times when window size changes.

→ align-self : centre
start
end

* align self : is (align items to flex containes)
↓ what
align self is align items for flex item.

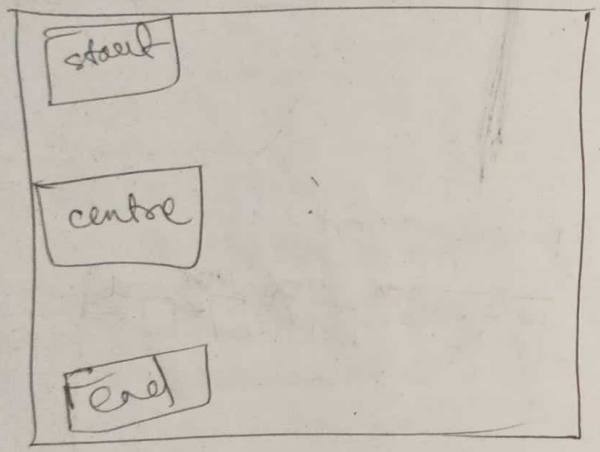
* {
margin : 0px
padding : 0px
- - - ;
}
will reset everything in css.

* align content : start | end | centre
x= x= x=
removes spaces at x and space between flex items.

* align-items : flex start
flex end
centre

~~align items~~ when flex is in row direction then align items works along ~~this~~ y axis.

align items works with crossaxis



* ~~stretch~~ : align-items : stretch

stretches flex items through the axis in which it is (cross axis)

* by default items will be set to stretch

* flex items properties

→ order : numbers
more orders, more last in line that flex item will be there.

→ flex ~~grow~~ shrink : numbers
shrinks or grows/expands numbers times when window size changes.

→ align-self : centre
start
end

* align self ... is (align items to flex containers)
↓
what ?
↓
align self is align items for flex item.

* {
margin : 0px
padding : 0px
- - - y
}
will reset everything in css.

* align content : start | end | centre
x= x= x=

removes spaces at x and space between flex items.

* Media query

① @media (min-width : xpx)
{
selector or element {
follows this condition until it reaches the min width mentioned above
}
}

② @media (max-width : xpx)
{
element selector {
follows this condition until it reaches the max-width mentioned above
}
}

③ we can also mention a range of screen resolution

@media (min-width : xpx and max-width : ypx)
{
element {
properties
}
}

* Implementing flex direction along with respect to the screen resolutions using @media query

* Box-shadow :-

i) Syntax : Box-shadow : x y Blur colour ;

ii) Box-shadow : x y Blur spread-rad (range of blur) colour ;

iii) Box-shadow : inset x y Blur colour ;
// to have shadow inside the box

iv) multiple shadows :-
Box-shadow : x1 y1 colour1 , x2 y2 colour2

* Note :- we can also create custom variables and assign values

Syntax :- --var-name : value ;

usage :- var (--var-name)

Ex :- --primary-colour : brown ;
background-colour : var(--primary-clour);

Note :- Scope of the variables is within the braces

* To define Global variables

Syntax: :root {

--primary-value : 330px;

--custom-colour : blue;

}

* text-shadow : x y blue colour;

Bootstrap

12 grid system.