5. CSS FLEXBOX & MEDIA QUERIES

Units:

Pixels:- Tiny dots on the screen. They are absolute unit.

↓

how many dots something should occupy horiztally & vertically

font-size: 16px

Make my font 16 dots high

Uses:

→To have more precise control.

Percentage:

To style an element relative to it's parent or containing element, this is used.

width: 80%;

↓

will take 80% of parent's width

Percentages for margin & padding are calculated based on the width of containing element.

Absolute: Independent. Irrespective of screen size or any other element.

Relative: Dependent. Related to othes elements.

Ex: Percentages

width: 50%;

↓

50% of containing element

Use case:

Responsive UI, where you want elements to change size based on screen or parent element.

ViewPort Width & ViewPort Height:

The entise web page that is visible is a viewport.

To style an element relative to viewport height/width this is used.

```
body {
    width: 80vw;
}
```

% → Relative to parent

vw/vh → Relative to webpage

REM:

Relative to the font-size of root element (usually the <html> tag).

font-size: 50 px;

↓

The font-size will remain

same even if screen size changes.

The size that look good in browser won't look good in tablet or phone.

Different size for different device.

REM is what is defined in html.

```
html {
  font-size: 16px
}

.main-content {
  font-size: 3rem -> 3 x 16
                  = 48px
```

}

Use: Great for consistent scaling across your website, as they always refer to single base size.

REM is a relative unit

Flexbox:

Flexible Box → Powerful tool for Responsive design

↓

It adjusts relative to the screen size.

Before flexbox we have media query.

↓

styling based on different screen sizes.

1000 – 1500px screen size

font-size: 15px

1500 – 1900px

- - - ..

Pain point to create responsive designs.

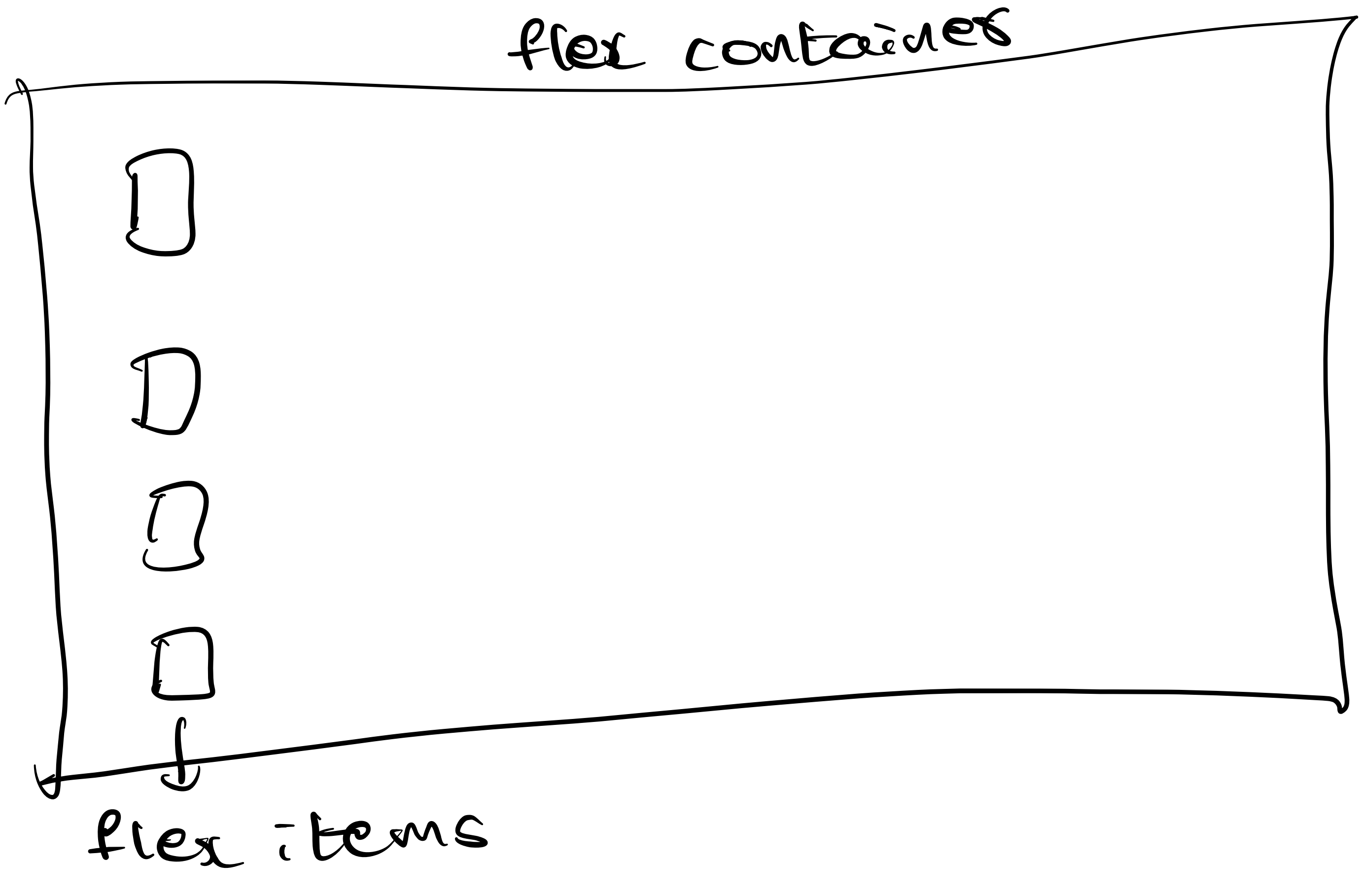
Flexbox : Layout model in CSS.

Makes arranging elements easier.

display: flex;

↓

You are creating a flex container.



Any children within the container are flex items.

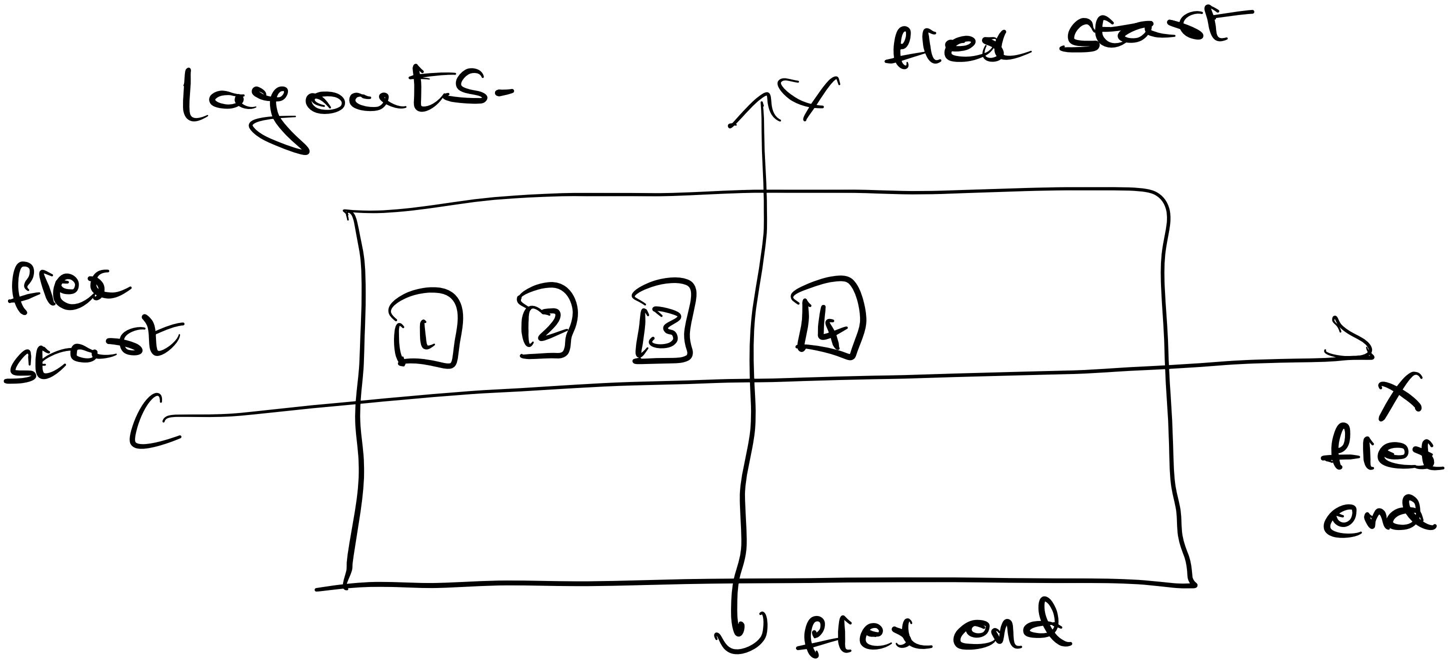
↓

follow rules & behavior defined by flexbox layout model.

Properties:

-> Flex items are arranged in a row.

-> Flexbox is used for 1 Dimensional layouts.



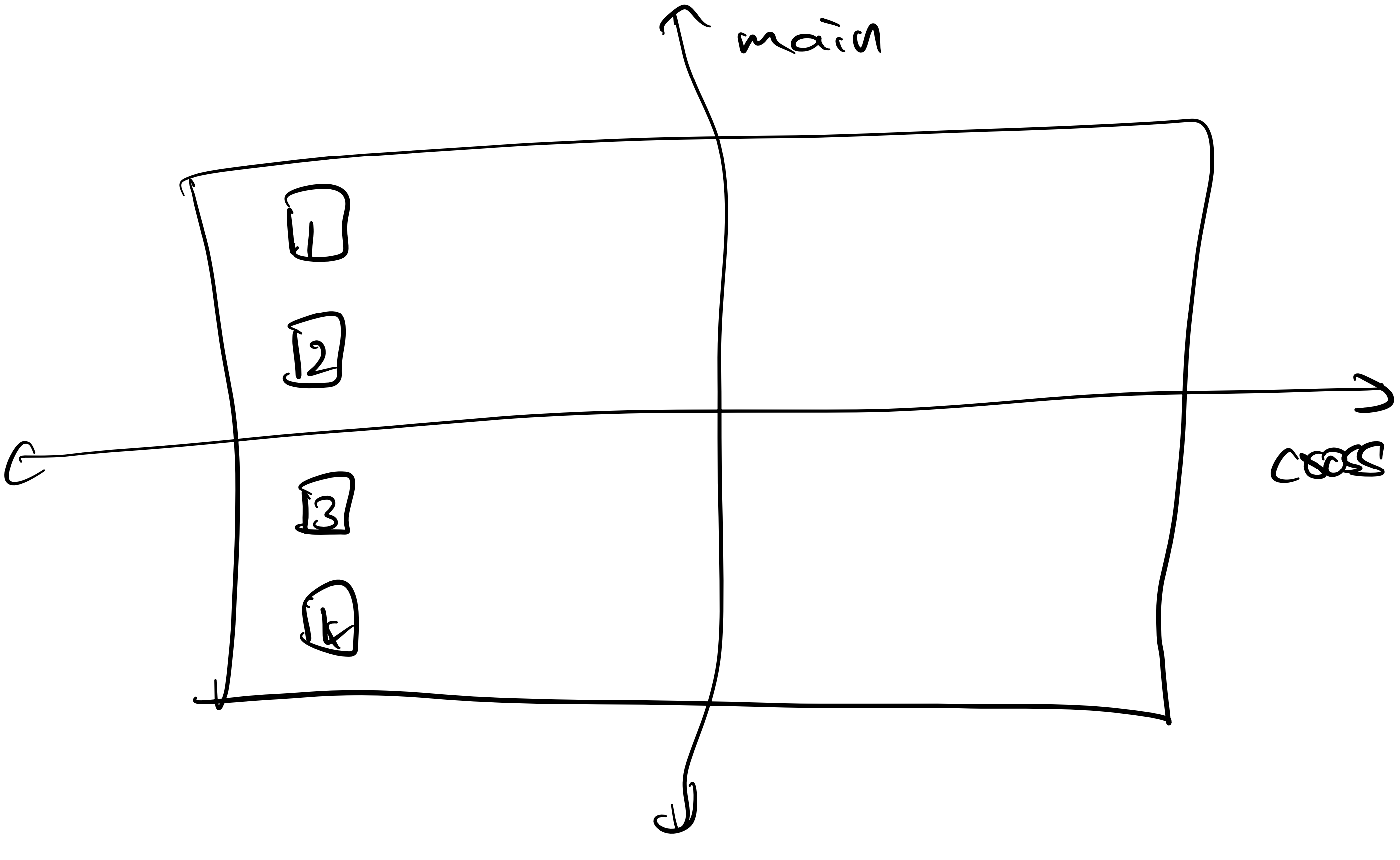
Main Axis → x (default)

Cross Axis → y (default)

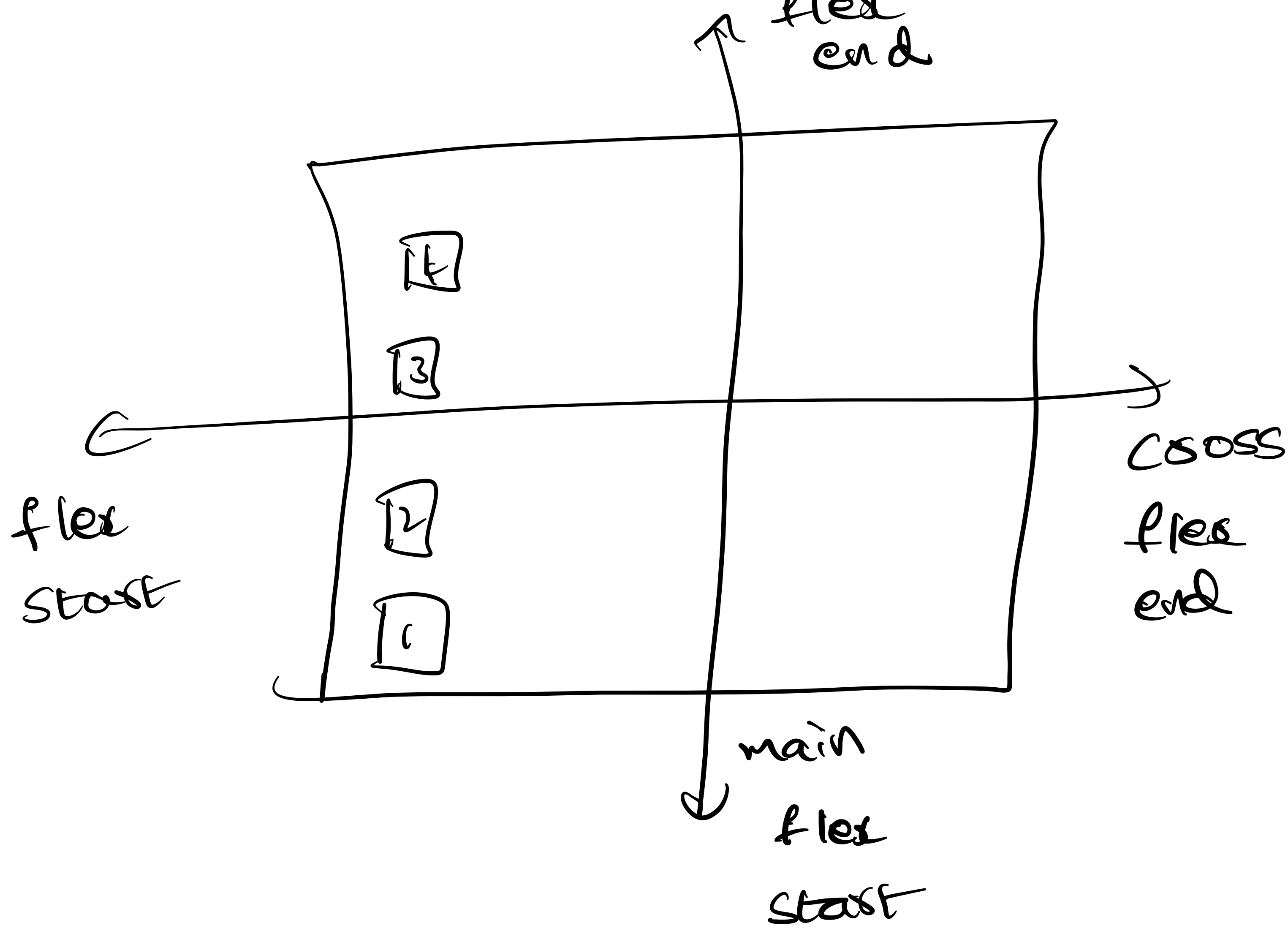
By default flex direction is row & elements are arranged on main axis.

what axis in main & what axis is cross is defined by flex direction.

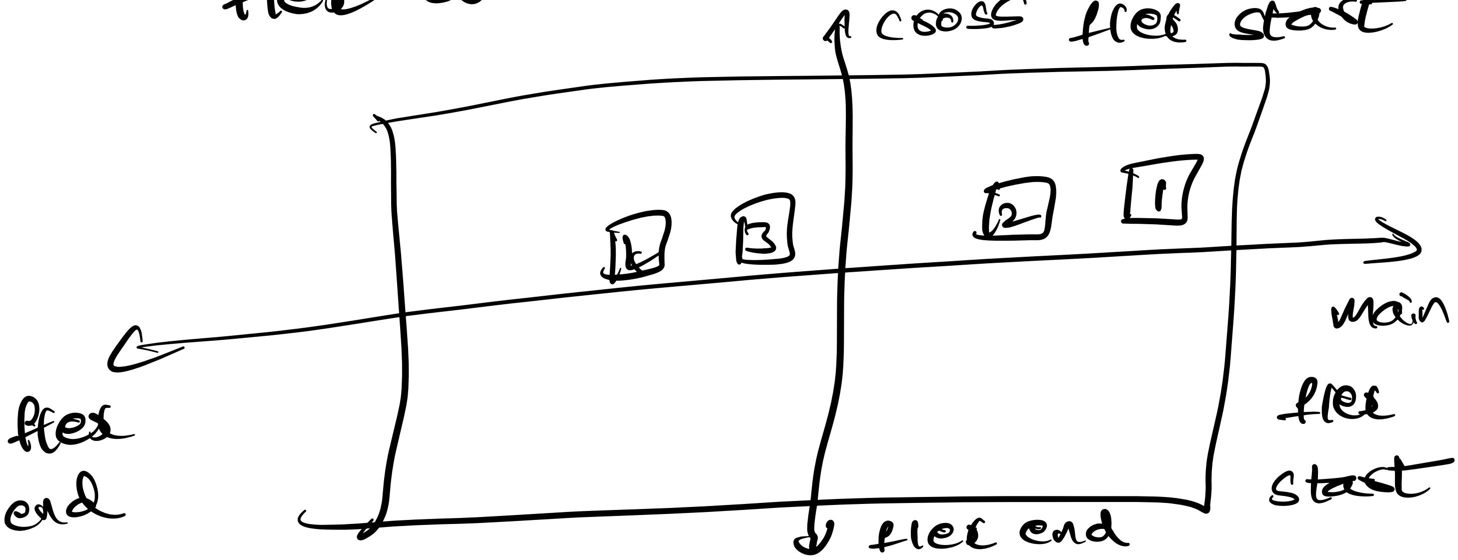
flex-direction: column;



flex-direction: column-reverse;



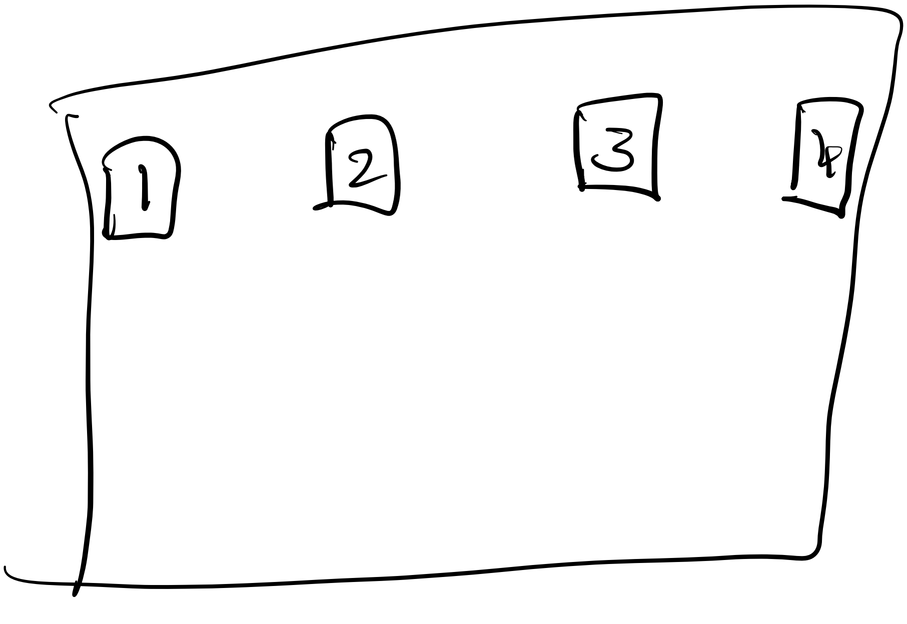
flex-direction: row-reverse;



justify-content : space between;

↓ to position flex items

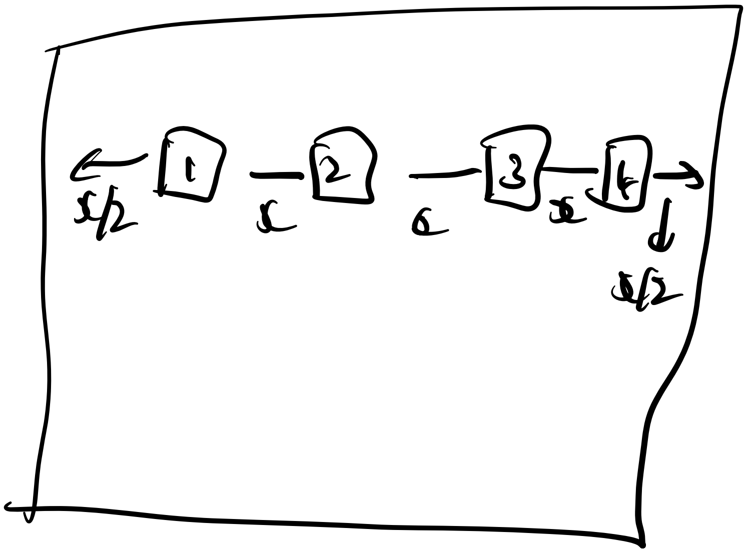
↓ No space at start & end but equal space between items.



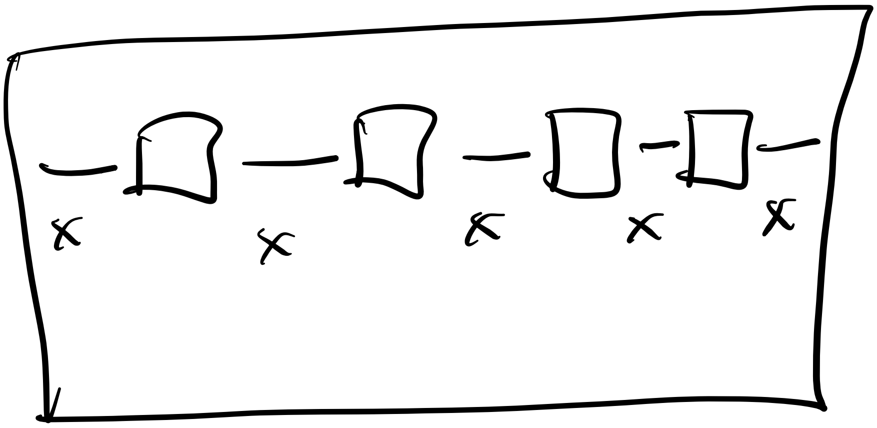
justify-content : space around;

space before & after first & last item.

Let's say space between items is x.



justify-content : space even;



All elements are evenly spaced

align-items : flex-start;

↓

To position items
across cross axis.

align-items: baseline;

↓

items are aligned across text base lines

Useful when items having varying font sizes or text content.

Even though px is absolute unit, when they are given for flex items, the items will shrink to accomodate.

To respect the values given,

flex-wrap: no-wrap;

↓

default value

items will shrink

flex-wrap: wrap;

Items will be added to new line