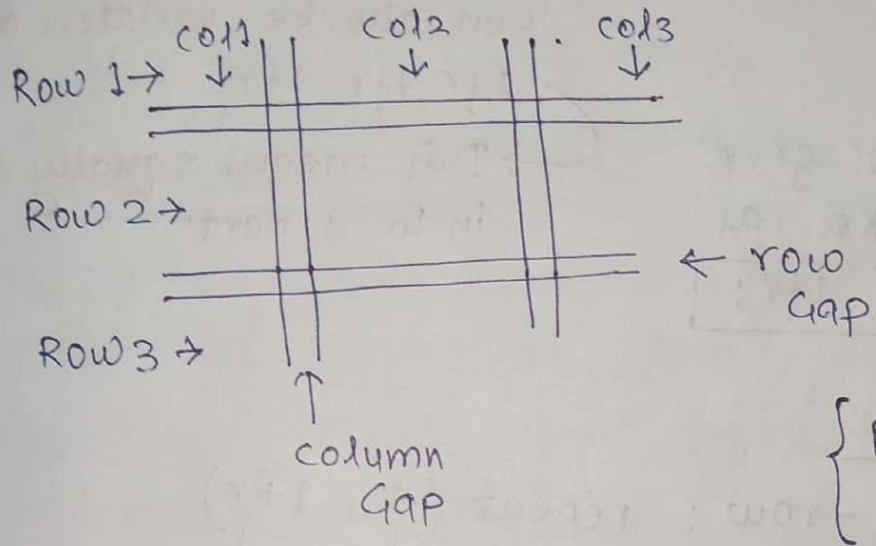
14/01/23

* CSS Grid

» A Grid is a collection of horizontal and vertical lines creating a pattern against which we can line up our design element.

» 2D layout, with rows and column and Gap

» This is matrix.



{ Flex Box - 1D
Grid - 2D

Ex column Gap

```
column Gap: 100PX;
```

Row Gap

```
row-gap: 50PX;
```

Shorthand property for Gap

```
gap: 50PX 100PX;
     ↑    ↑
    row  column
```

* Display Property

```
.grid-container {
    display: grid / inline-grid;
}
```

* Grid Container

Grid container is a parent container that cont
of grid items (child container) in row and colu
format.

① For Grid column

grid-template-column : repeat (3, 1fr)

(or)

can also be written as
1fr 1fr 1fr

→ This means equally divide
in to 3 part.

we can also give
value in pixel as

```
200px 250px 1fr;
```

② For Grid Row

grid-template-row : repeat (4, 1fr)

4 time — Equally divided

③ For Gap in Grid

Gap : 15 PX;

④ Line-Based Placement

Grid item property

ⓐ grid-column-start
ⓑ grid-column-end
ⓒ grid-row-start
ⓓ grid-row-end
ⓔ Shorthand Properties:- grid row, grid column, grid area;

⑤ Justify-Content Property (horizontal axis)

* start
* end
* center
* space-between
* space around
* space-evenly

⑥ Align-content Property

more than one row

* start
* end
* center
* space-between
* Space-around
* Space-evenly

* Grid-Item

① grid-column property

The grid-column property defines on which column to place an item.

Ex ⇒ Make 'item 1' start on column 1 and end before column 5.

```
.item 1 {
    grid-column : 1/5;
}
```

```
.item 1{
    grid-column-start: 1;
    grid-column-end :5;
}
```

make 'item 1' start on column 1 and span 3 column.

```
.item {
    grid-column : 1/span 3;
}
```

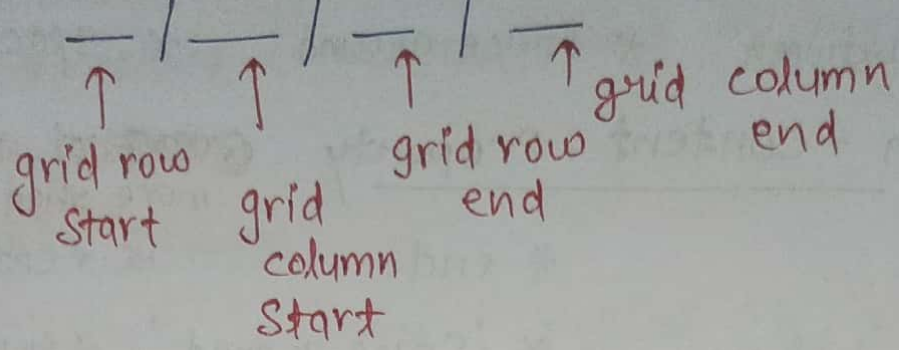
② grid-row-property

This property defines on which row to place an item.

Ex ⇒ Make 'item1' start on row-line 1 and end on row-line 4.

```
.item {
    grid-row : 1/4;
}
```

* Grid area — Shorthand



The grid-area property specifies a particular area or set of rows & column that a grid item occupies. It is applied to the grid item itself with CSS.

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

* Grid-template-area

grid-template-area is the property used to name the rows and column of a grid & to set its layout.

```
.container {
    display : grid;
    grid-template-column : 300px 300px 300px;
    grid-template-rows : 250px 30px;
    grid-template-area : "hd hd hd hd hd hd hd hd"
                         "sd sd sd main main main main main"
                         "ft ft ft ft ft ft ft ft";
```

```
.header {
    grid-area : hd;
}
```

working

	col 1	col 2
row 1	hd	hd
row 2	side	side
row 3	ft	ft

← This is how naming is done & when we give value that occupies the space named

Q:) If we have rows and column defining in grid & can name them with (grid-template-area) & then give more data (box). than define what will happen to the extra box

↳ They are all squeeze below the container.

↳ eg =

```
#header{
  grid-area: hd;
}
#footer{
  grid-area: ft;
}
#sidebar{
  grid_area: side;
}
```

* Advanced grid concept

① Fr unit — box are equally divided using 1fr.

② Repeat function repeat (3, 1fr)

③ Grid-auto-rows : minmax ( )

↑ when there are unknown numbers of rows we use this property.

↖ minimum & maximum size of the row.

eg ⇒
grid-auto-rows:
minmax(100px, auto);

④ Grid-auto-column : ?

* Grid Properties

① Grid container Property

ⓐ justify-content (horizontal axis)

* start
* space-between
* end
* space-around
* center
* space-evenly

ⓑ align-items (vertical axis)

* start
* stretch
* end
* baseline
* center

② Grid items Property

ⓒ justify-items
changes width according to content (horizontal)

* start
* end
* stretch
* center
~~* space-between~~
~~* space-around~~
~~* space-evenly~~

ⓓ align-content
change width according to content (vertically)

* start
* space-around
* center
* space-between

ⓔ align-self (vertically)
* stretch
* center
* start
* end

ⓕ justify-self

* stretch
* center
* start
* end

change width a particular box. (horizontal)

ⓖ place-items
ⓗ place-self
} shorthand property

* Media Queries

Responsiveness css style in every device.

1) "viewport" → The area of the window in which web content can be seen.

2) Media Queries are used to set different style rules for different devices or sized screen. we use breakpoints to set the condition of a media Query.

@media (feature : value)

3) Essentially media Query breakpoint are pixel value that a developer /designer can define in CSS when a responsive website reaches those pixel value, a transformation (such as the one detailed above) occure so that the website offers an optimal user experience.

for Muliple Breakpoint

we use (and) Keyboard.

```
@media (min-width : 600px) and
       (max-width : 900px)
{
    .container {
        // rule to change size;
    }
}
```

* Nested Grid

Is simple and can be done simple by using the display : grid rule for both a parent & child elemnt

```
.container {
    display : grid;
}
```

```
## one {
    display : grid;
}
```