CSS units (measurements):

1. Absolute Unit : Non-dependent on something
   ↳ (px)

2. Relative Unit : Dependent on something
   ↳ (em, rem, %, vh, vw, ch)

⟹ default (px)

How to choose units?

- px : Pixels, absolute length
- em : Relative to font size
- rem : Relative to root element font size
- vw : 1% of viewport width
- vh : 1% of viewport height
- ch : width of "0" character (<p>)
- % : percentage, usually relative to the root value.

16px

Padding : 10em;
: 10vh;
: 10vw;

width/height :

- % to be relative to parent
- vw/vh to be relative to viewport
- ch for paragraph widths
- rem for closer to absolute values
  - px true absolutes

Margin / padding:

- rem for closer to absolute values
- em to scale with font size
- px for small values

rem / em

Welcome!!

Borders, Shawdows

- px for small values
- rem/em optional (doesn't always look great)

Font Size -

- rem is usually best
- em to scale to parent size
- px last option ✓

Colors

- Keywords: red, blue, green
- Hex RGB: #4B70AF

red → 4B, green → 70, blue → AF

rgb(75, 125, 175)

red, green, blue

Position

- Static : default
- fixed : The element is positioned relative to the viewport and removed from normal document flow.
- relative : The element is positioned in the same place according to the normal document flow.
- Sticky :
- absolute :

Sticky:

- The element will act similar to a 'relative' positioned element
- But once it scrolls off screen it will stay fixed to the screen [display: fixed]

Absolute

- This will act same as 'fixed'
- Except the element will be positioned relative to document flow
- Oftentimes referred to as an ancestor (parent), has a position value other than "static", then it will be positioned relative to that nearest positioned ancestor!)

Flexbox:

→ A layout model, also known as flexible Box Layout module

→ Particularly useful for building responsive designs

→ Row and column layout

→ An element can be made a flex container with "display: flex" → direct children (flex items)

```
<section>  (flex container)
  <div>  (flex items)
    <p></p>
    <p></p>   } grandchildren  3
  </div>
  <div></div>  (flex items)
  <div></div>  (flex items)
</section>
```

```
section {
  display: flex;
```

Some of the most common properties used to layout flex items are:

* flex-direction : row / column / row-reverse / column-reverse
* justify-content : flex-start / flex-end / center / space-around / space-between / space-evenly
* align-items : flex-start / flex-end . . . .
* flex-wrap : wrap / nowrap / wrap-reverse
* align-content : . . . . .
* flex-flow : flex-direction / flex-wrap
* gap : . . . . .

Flex-items:

- → align-self
- → flex-basis
- → flex-shrink
- → flex-grow
- → flex
- → order

CSS Grid:-

- → A layout model for creating responsive design
- → Multiple rows & columns (2 dimensional)
- → An element can be made as grid container with "display: grid", all its direct children will be its grid items!

grid-template columns:

grid-template rows:

grid-template areas:

justify-content, align-content, align-items,

justify-items, place-items, gap

grid-column, grid-row, grid-area

grid-container

grid-items

Mobile First Approach:

→ 1st preference should be given to mobile view

body {

"h h h
m a n
f f f"

3

80 px
body {
(Mobile) "h
n
m
a
f"

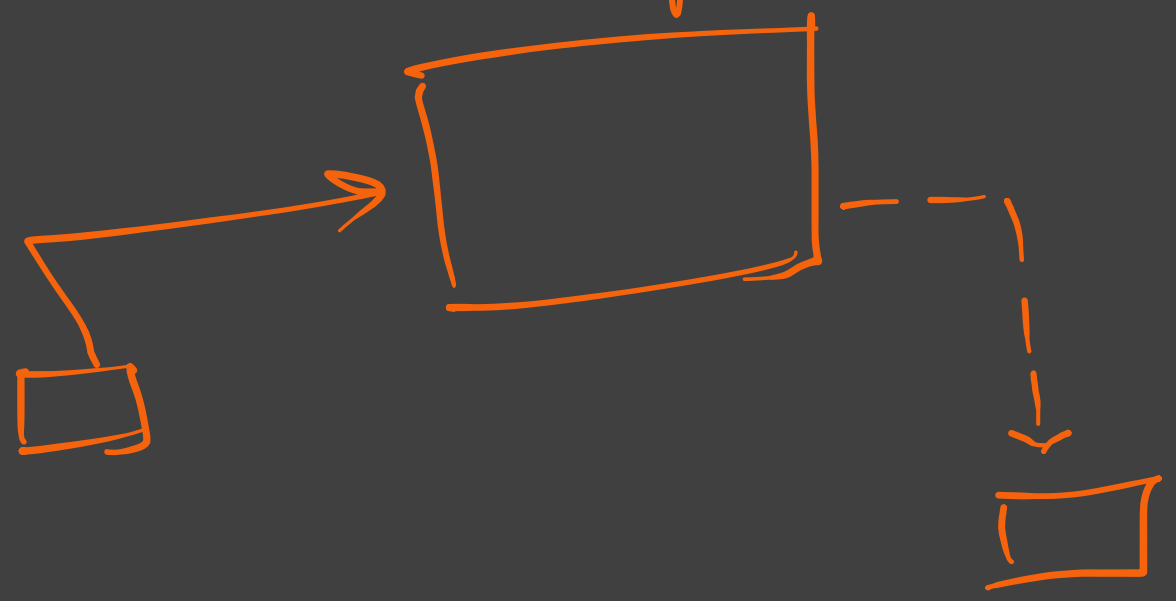
3

1st

600 px
body {
"h
n
m
f"

3

Mobile first Approach!



CSS - overflow :

"Content" - "overflow is a property that controls what happens when the content of an element overflows its containing box"

- Visible : Overflows the container
- hidden : content is not visible
- scroll : scroll bar will be added
- auto : overflow is there, scroll bar will be added else normal!!

Bootstrap:

- How to set up bootstrap?
- What is the layout?
- What is container, container-fluid, container with breakpoints.
- What is a grid in bootstrap?
- column prefix & its use and max columns for a row?
- Nested columns & column ordering!

HTML, CSS

- styles, properties, selectors
- responsive [Grid, flex, media query]

CSS library for RWD - Responsive Web Design

Bootstrap: To build fast and responsive web sites/application.

CSS library!

↑ less code

1. How to Set up Bootstrap?

- using CDN (Content Delivery Network)
- download and refer

2. Layout (predefined classes)

- Responsive, mobile-first fluid grid system that scales upto 12 columns as the device/view port size increases.

3. Container:

- It is a centered, fixed-width.

container-fluid:

- Viewport entire width will be occupied.

Container with breakpoints:

- provides containers that have a maximum width for specific breakpoints, to make the more responsive.

What is grid in Bootstrap?

rows - parent
col - immediate children

- Series of containers, rows and columns
  - ↳ content alignment!
- Rows: used for creating horizontal groups of columns

1	2	3	4	5	6	7	8	9	10	11	12

→ col-12
→ col-md-4

} Columns (actual content containers) (content should be placed within the column)

Column prefix:

- .col-12
- .col-sm-4
- .col-md-5

WINDOW - Common util functions:

1. Getting URL information
2. Setting URL
3. Opening a new window
4. Checking window width
5. Scroll to top

6. SetTimeout ✓

7. SetInterval ✓

8. localStorage - SetItem, getItem ✓

9. sessionstorage - SetItem, getItem ✓

Q & doubts?

→ localStorage [long lasting]

→ Session storage [temp - browser close]

Add to do

① SaveLocalStorage (setItem) — tasks = [0{text: '—', completed: true},
1{ — — — }
2{ — — — — }];

② Load (getItem) (every time we refresh the page)

③ Create task element

for (task)
{ task
}