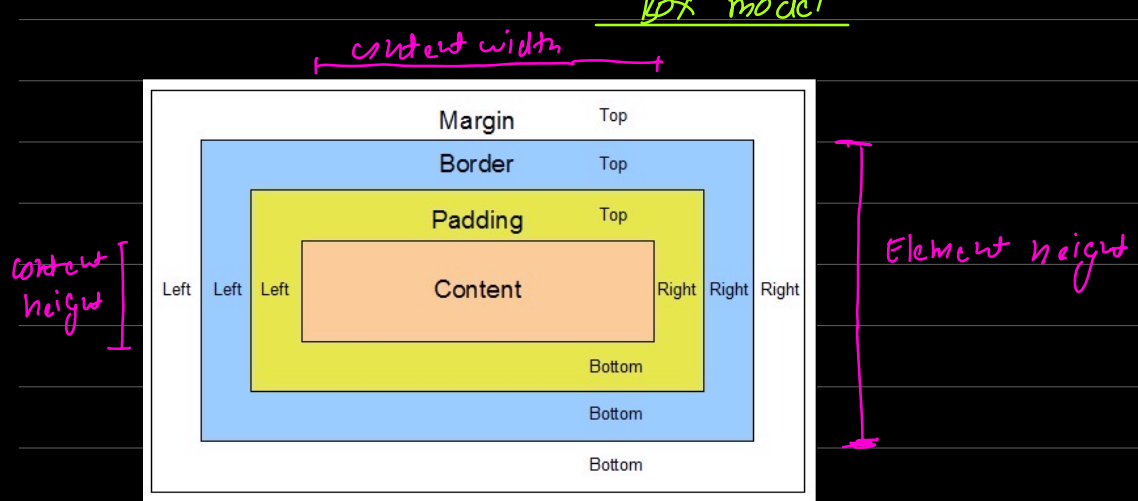
Agenda

1. CSS - refresher
   → box model
2. Overflow
3. display (block, inline, inline-block)
4. Units
   → absolute
   → relative
5. Specificity, inheritence
6. project / positioning

Box model



Element = content + padding + border

default
↳ (Height / width) refer to content

Border-Box (box-sizing)
↳ (Height / width) refer to element

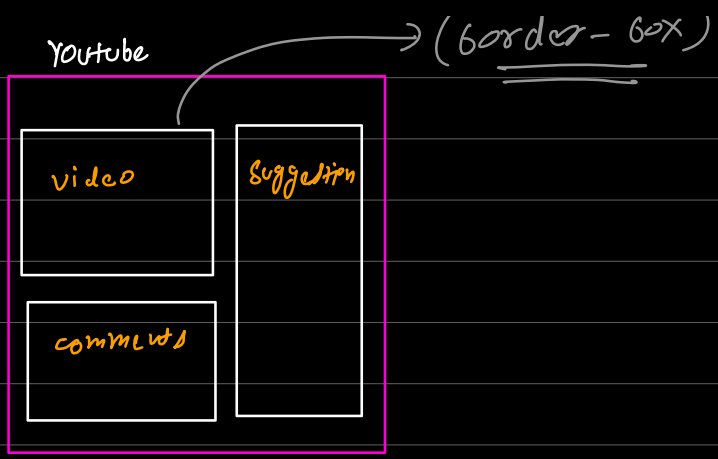
(·) By default, padding and border is 0.

$Element_{H/W} = content_{H/W}$

(Element) = (content) + (padding) + (border)
(use-define) (dynamic) (use-define) (use-define)

Q [100, 100) = [?, ?] [30+30) + [5+5]

[C_H, C_W] = [30, 30] (100-60-10, 100-60-10)



overflow

(height of the Element)

→ (default)

→ (with content) (H/W → content/H/W)

→ (no-content) (H/W = 0)

→ (user-define) → Height/width given by user.

[H/W] of content > (H/W) of box

types of overflow

- visible : (default behaviour)
- hidden : (extra content will get hide)
  (using js we can provide styling for hidden content)
- clip : (Extra content will be deleted forever)

Analogy

hidden : extra content is in recyclebin
clip : extra content is permantly deleted.

- scroll : (add scroll on both side)
- auto : (add scroll on both side as per the needs)

display

block : (H/W) as much as possible
Next element will start from next-line
Example: div, p, semantic tags.

inline : → default H/w of content, I can't provide user-define H, W.

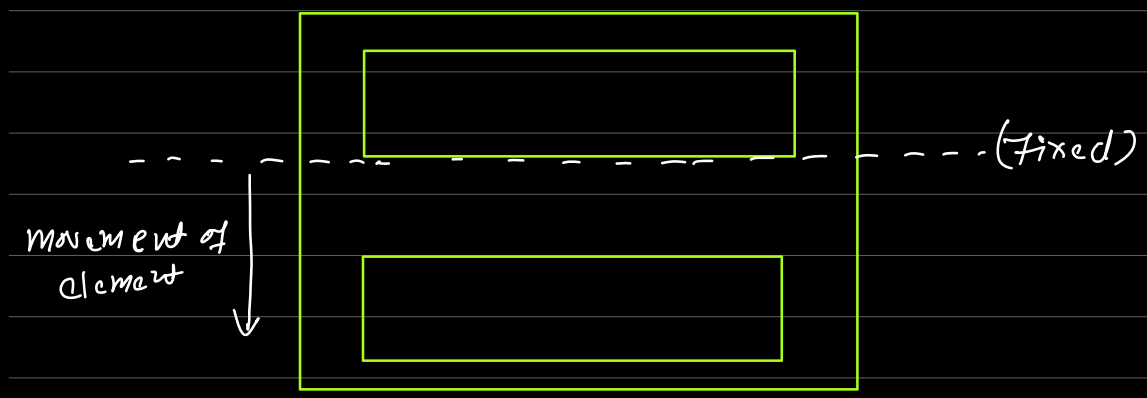
→ Next element will start from right (just next to current element)

Ex: anchor, span.

→ inline element can overlap (invade) to some other element.

inline-block

↳ H/w is user-define next element, same as inline



Units

- Absolute : px
- Relative : percentage, rem, em, vh-vw

(*) By default 16 px is size of text.

Specificity

→ narrow down your search region (How much specific / clear you are)

→ p is generic

→ class help me to narrow down

→ id is unique to more specific

→ if some information have same priority.
then order matters.

inheritance < p < class < id < inline < important.