


full stack development

Internet



mobiles
desktop apps
web apps
cars
TV / watches

api's
databases
reliability
security
platforms

→ CLI → command line

linux/mac

→ * GUI *

fast

↳ server side

easier to automate

Shell

Command interpreter that helps us to interface
with computer system: → system calls → fork

Shell → application → kernel

How Internet works 2.2

Internet is a system of globally interconnected devices.

→ 2016-17 Approx 4.5 billion + users.

> 50% of the world population

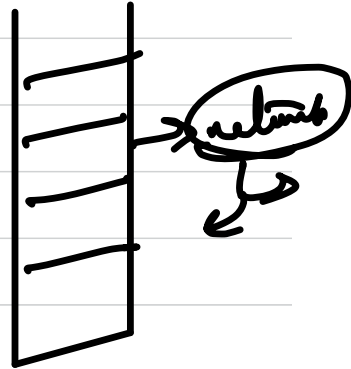
1.5 billion + websites

10x increase in the internet usage from 1999 to 2013

→ [www.google.com] → ip address

DNS (Domain Name Server)

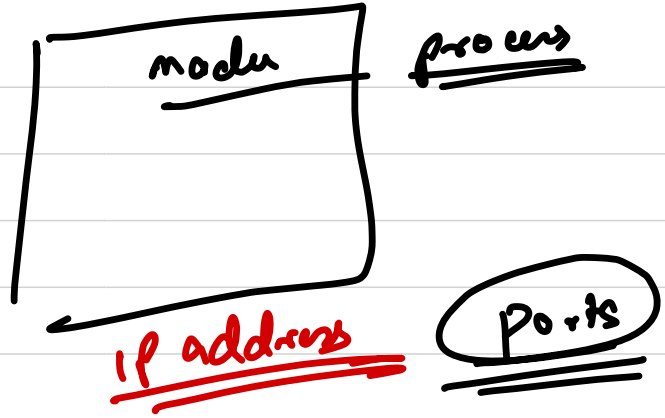
↳ phonebook of internet



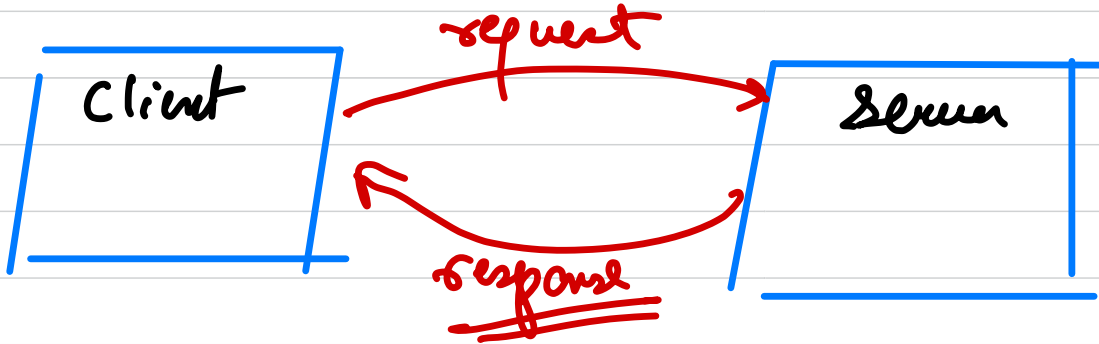
ISP (Internet Service Provider)

↳ route

Data center



packets



protocols

HTTP

FTP

HTTPS

SMTP

etc...

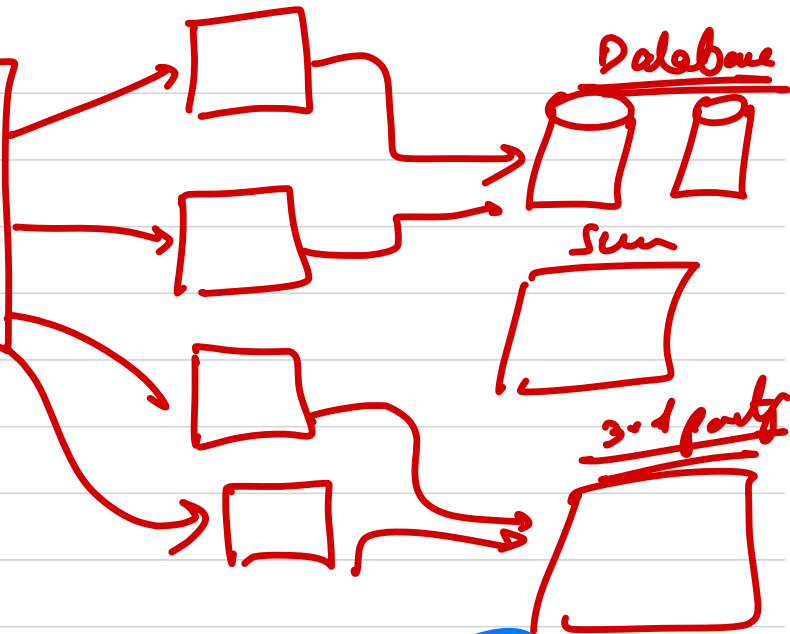
Components of Web

- 1) Web page → single document (HTML)
- 2) Web server → hardware or software that hosts websites.
- 3) Search engine
- 4) Website.

Client



LOAD
BALANCE



Database

Server

3-4 parts

login → fb
google

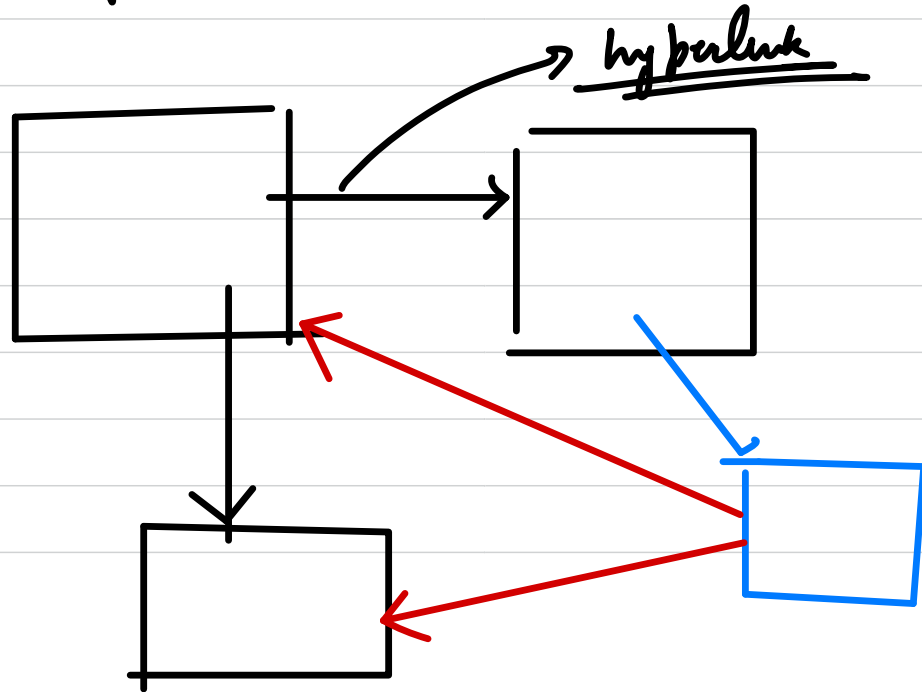
Client

HTML

[≡] web page
 mobile
 desktop → HTML (Hyper Text Markup Language)

Hyper text is a special text that contains links to other text.

(xml)



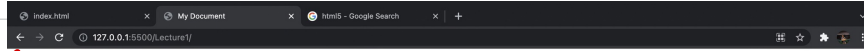
XML \Rightarrow Extensible Markup lang
 \hookrightarrow tag based language

<age> 23 </age>
 \downarrow \downarrow
opening closing

one html tag \rightarrow one html element

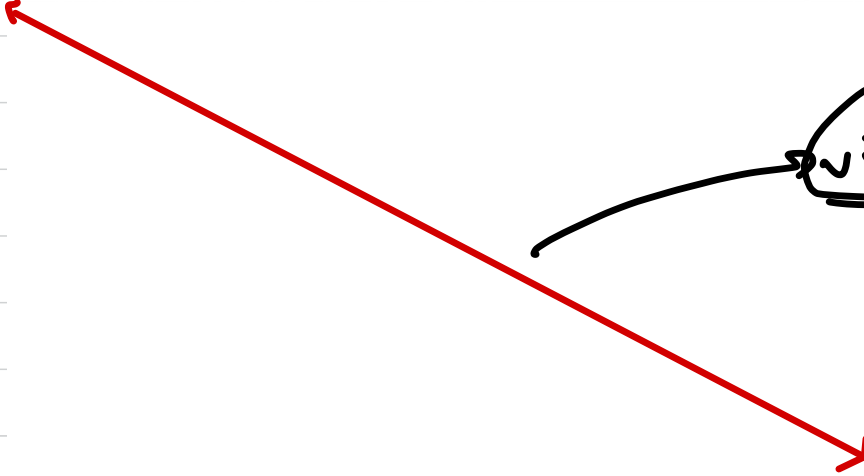
every element has properties \rightarrow attributes

$\left[\begin{array}{l} \text{HTML} \rightarrow \text{Skeleton} \\ \text{CSS} \rightarrow \text{Beauty} \\ \text{JS} \rightarrow \text{motion} \end{array} \right] \rightarrow \underline{\underline{\text{Live}}}$

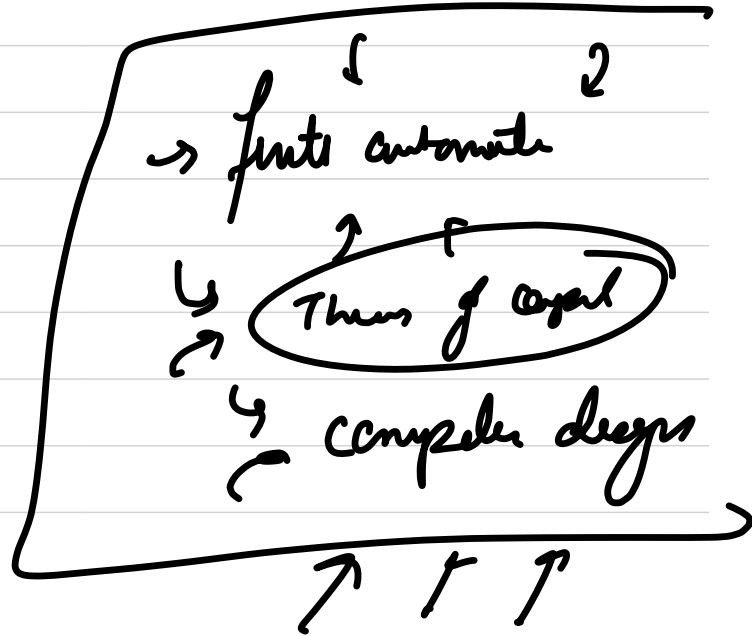


white
space is the
viewport

viewport



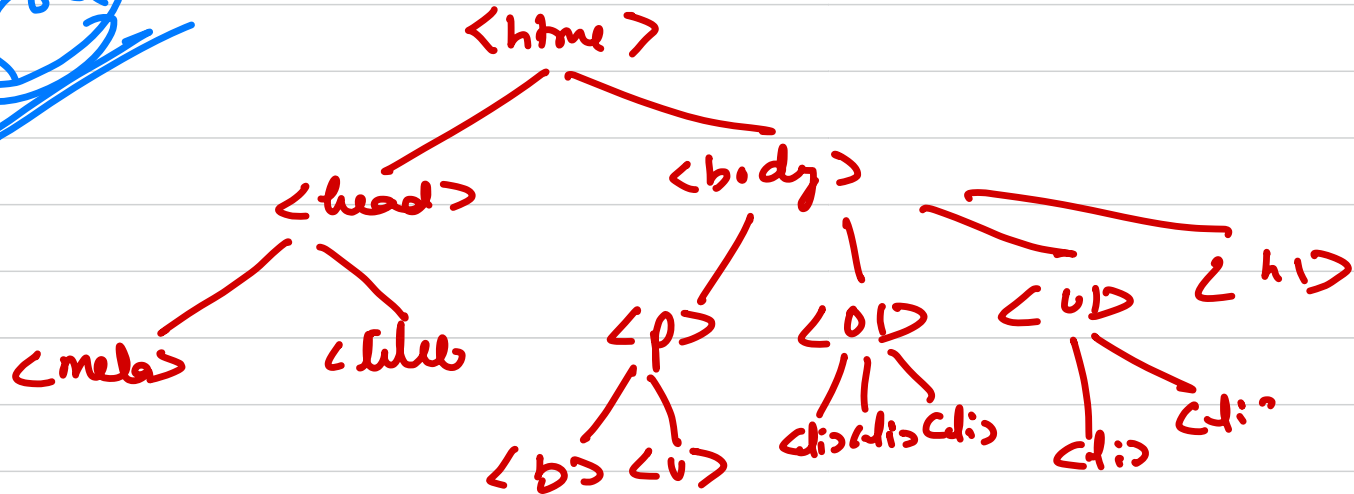
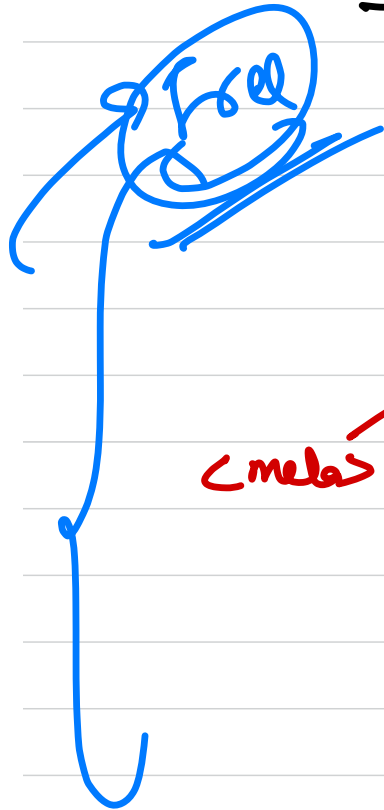
even if CSS and JS is disabled, we should
structure html such that we are able to depict
all info properly



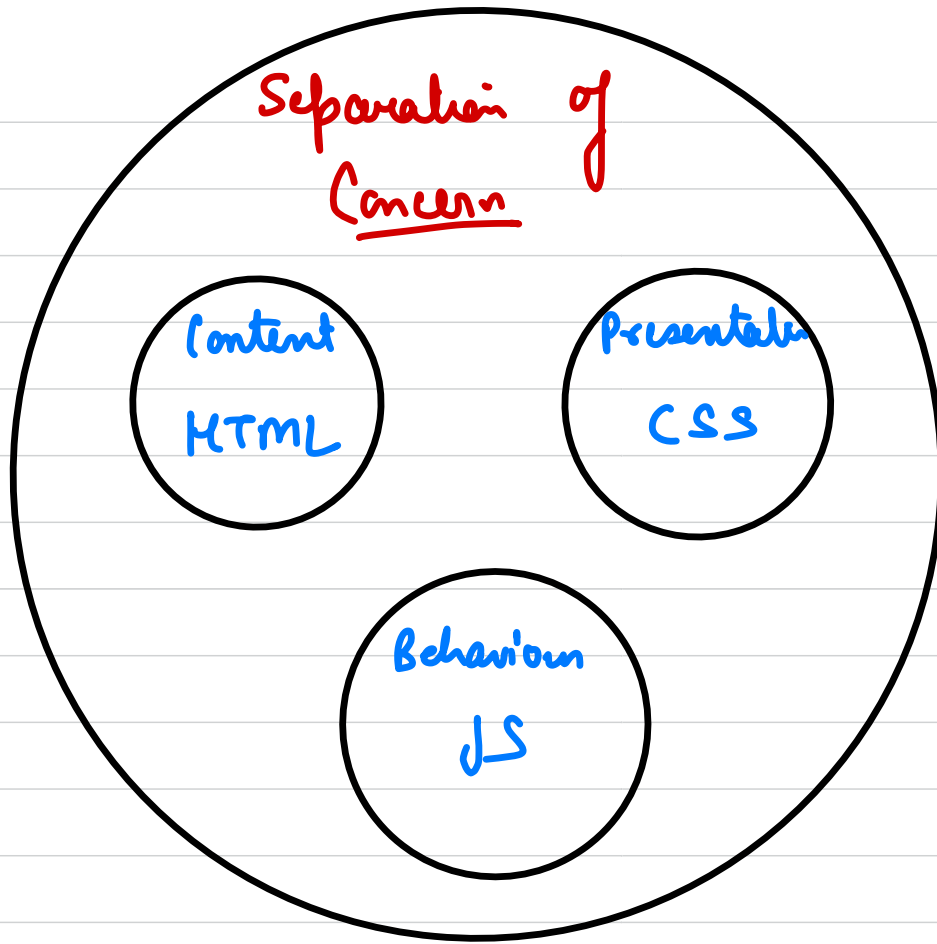
→ Block Tag → h_1, \dots

→ Inline Tag, span

DOM → Document Object Model



Cascading
Style
Sheet



CSS tells
the browser
how HTML
structure
is to be
presented

How to add CSS in HTML DOC??

Can cause
performance
issue

External

↓
We put CSS in
another file & link
that file to HTML

Embedded

↓
Write CSS in
HTML DOC

inline

↓
add CSS
directly to
an HTML Tag

`< p style = " color : black ; " > Content < / p >`

HTML element
attribute

Inline CSS

Code
CSS file → .css

(link this file) → External CSS

`< link href = ". / style / 1 . css " rel = " stylesheet " / >`

→ one of the best
ways

Embedded CSS

```
<style>  
  {  
    =  
  }  
</style>
```

External Stylesheet →

→ reusable

maintainable

Cachable

code separation

Changes are circumvented

* Selectors → these are used to refer to exact html elements. These selectors are used for referring individual & group of elements.

`selector {
 - property: value;
 ==
}`

→ Basic CSS

`<id="" class="" >`

→ There are 3 basic selectors, → CSS / JS

id selector → `#myid {`
`}`

class selector → `.class {`
`}`

tag selector → `div {`
`}`

Initially



```
tag1 { }
```

```
Tag1, Tag2 { }
```

```
.class { }
```

```
# 10
```

```
: link
```

```
: action
```

latu

$\begin{matrix} E & \searrow & F \\ E & > & F \\ E & + & F \\ & \downarrow & \end{matrix}$

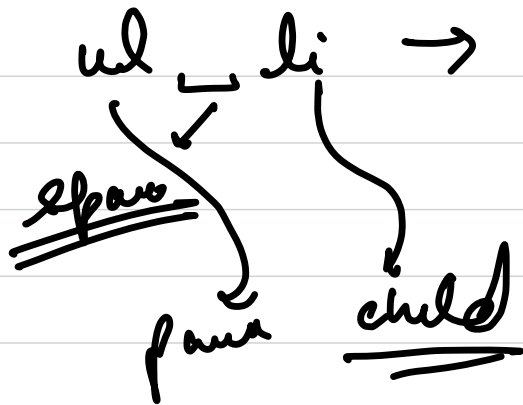
:first-club

Specifying

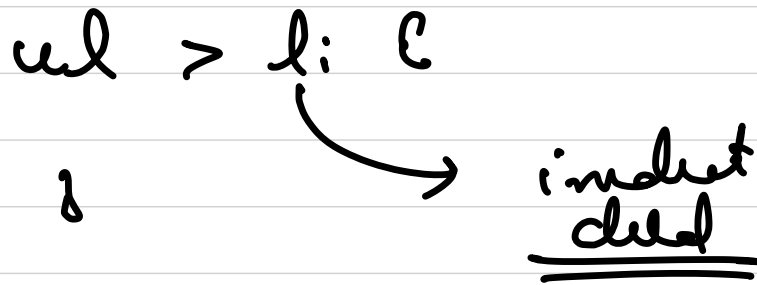
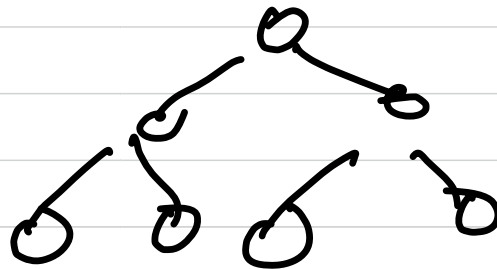
0 0 1 \rightarrow tag

0 1 0 \rightarrow class

1 0 0 \rightarrow ID



descendant selector



$X + Y$

\downarrow
 Y that immediately follows X .

$\begin{matrix} & 2 \\ & \swarrow \searrow \\ X & \sim & Y \end{matrix} \rightarrow$ sets the CSS for all Y elements
that are preceded by a X
element with same parent.
 $\{$
 $color: blue;$
 $\}$

* Attribute selector \rightarrow they have same specification
as class



$E[attr]$ \rightarrow element E that has attribute $attr$ with any value

$E[attr = "val"]$ \rightarrow element E that has attribute $attr$ with value val . case sensitive

$E[attr | = val]$ \rightarrow Element E whose attribute $attr$ has a value val or begins with "val"

$$\Sigma [attr = val] \rightarrow \text{Element } E \text{ whose attribute}$$

Starts with val.

``

$\langle a \text{ href} = "https://__" \rangle$

$\angle a \text{ hy} = "http://\underline{\hspace{1cm}}"$

$m \in \Phi \rightarrow$
 $\langle \text{option} \rangle^2$
 $\hookrightarrow \hookrightarrow \vdots \hookrightarrow$

$\exists [\text{attr} = \text{val}] \rightarrow \text{element } e \text{ whose attribute}$
 end with val

$\varepsilon [\text{attr}^* = \text{val}] \rightarrow \text{element } \Sigma \text{ whose attribute}$
has a substring val.

$\Sigma [attr = " " i]$

can insert

UI pseudo class

: enabled

: disabled

: checked

structural selector

: root \rightarrow It selects the root element.

: only-child \rightarrow It targets an element that is only child of its parent. This means parent contains only one element.

: empty \rightarrow

: n^{th} -child

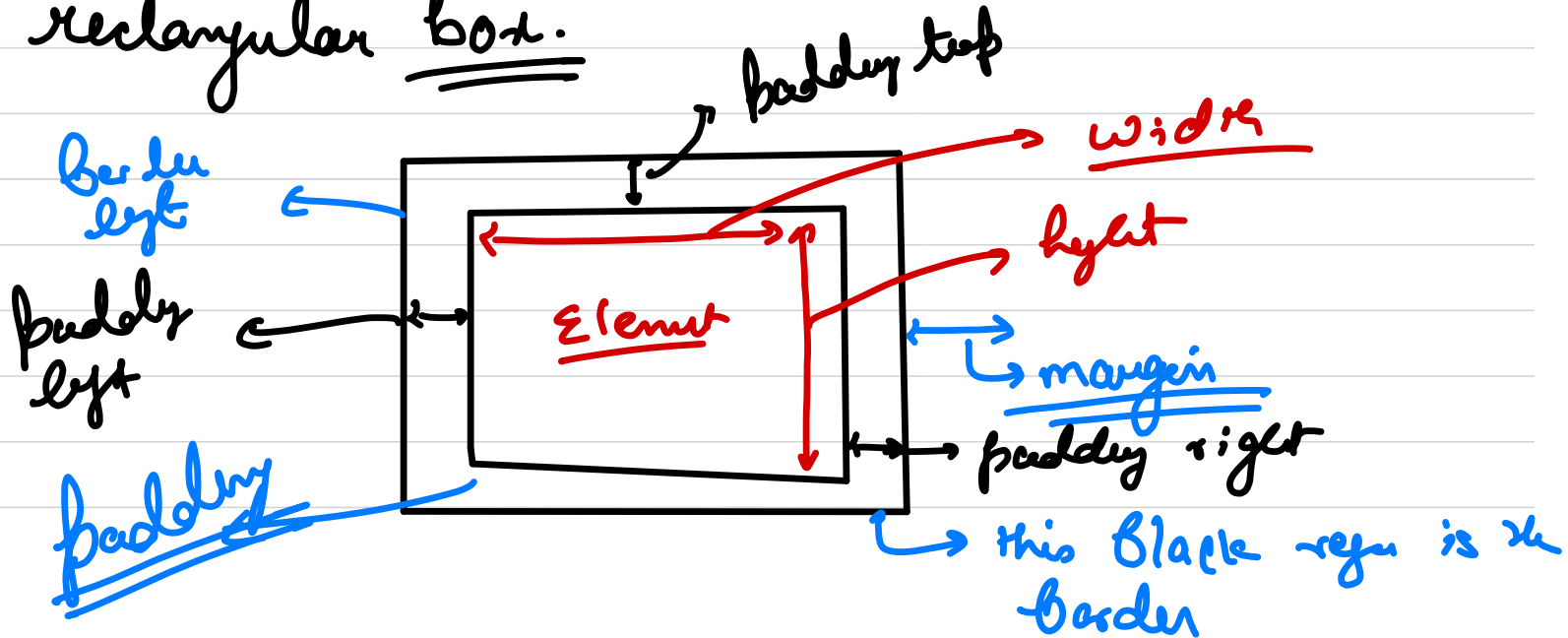
Counters in CSS

↳ we can change page appearance based on location on a doc.

→ property → counter-reset → (0 default)

Box Model

↳ every element in the web design is a rectangular box.



Width \rightarrow $el_width + padding\ left + padding\ right$
 $+ border\ left + border\ right.$

height \rightarrow $el_height + padding\ top + padding\ bottom$
 $+ border\ top + border\ bottom.$

if we don't define padding & border \rightarrow then it
will be either zero or browser default.

CSS-positioning → position

static

↓
position: static

↙
default

relative

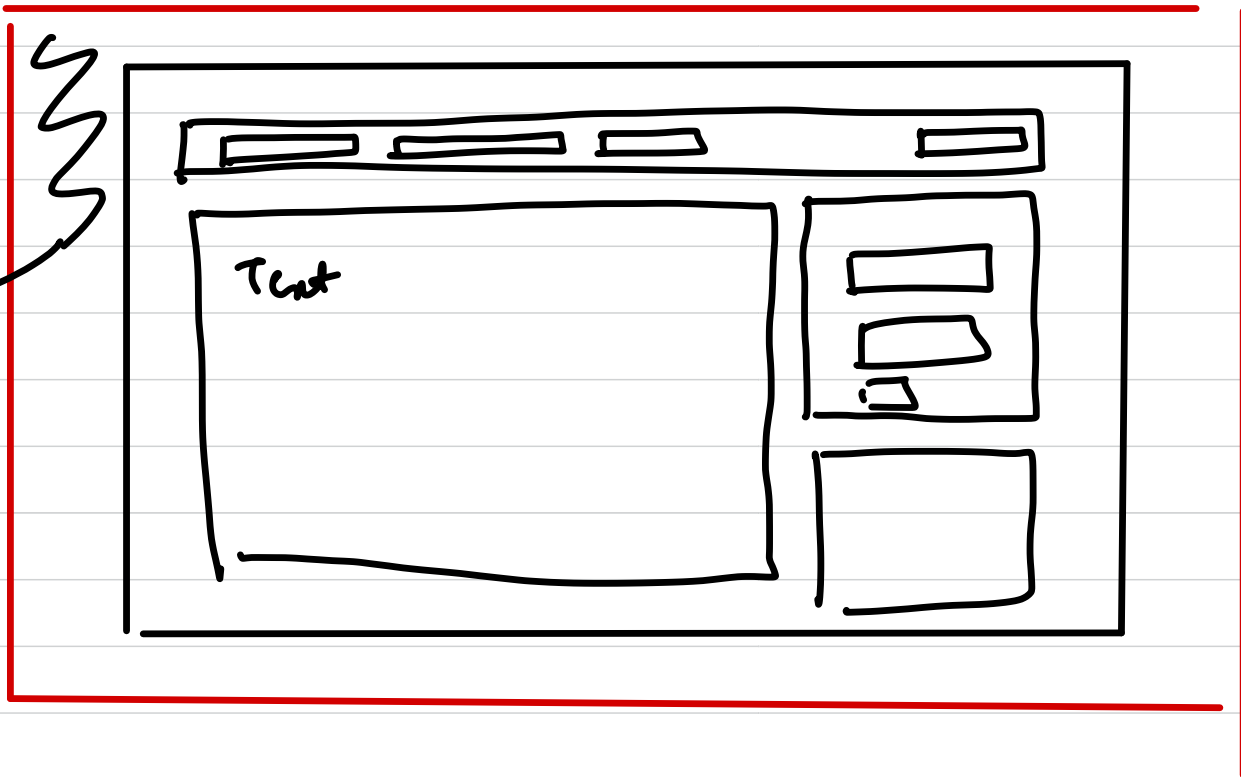
→ absolute
→ relative
→ fixed

Static positioning → default positioning of every element regardless where we declare it.

Relative position → It helps the element to be offset relative to themselves, parents or viewport.

→ If we don't give a middle, & box has
static or relation position, middle remains
loose, padding & border will push elements
inside of outward

blue



flim box