

## Lecture : CSS

### Cascading style sheet

`<h1 style = " color: blue ; text-align:center;" >`  
↑  
attributes of h1

Colours Hex Value (RGB)

12, 142, 5  
↓  
# 0c 8e 05 (base 16)

color: # 0c 8e 05

Keep CSS elements (styling)

separately from actual content of webpage

one way

`<html>`

:

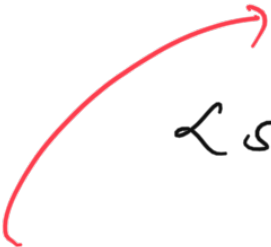
add a `< style >`

`</style>` in `<head>` `</head>`

Inside `<style>`

```
h1 { color: blue;
    text-align: center;
}
```

`<style>`



wherever  
you see h1,  
use this style.

maintain a separate style page!

Add a `<link rel="stylesheet" href="styles.css">`

In `<head> </head>` in your html page  
`styles.css` is your css file

styles.css

```
h1 { color: blue;
    text-align: center;
}
```

Other properties you can specify

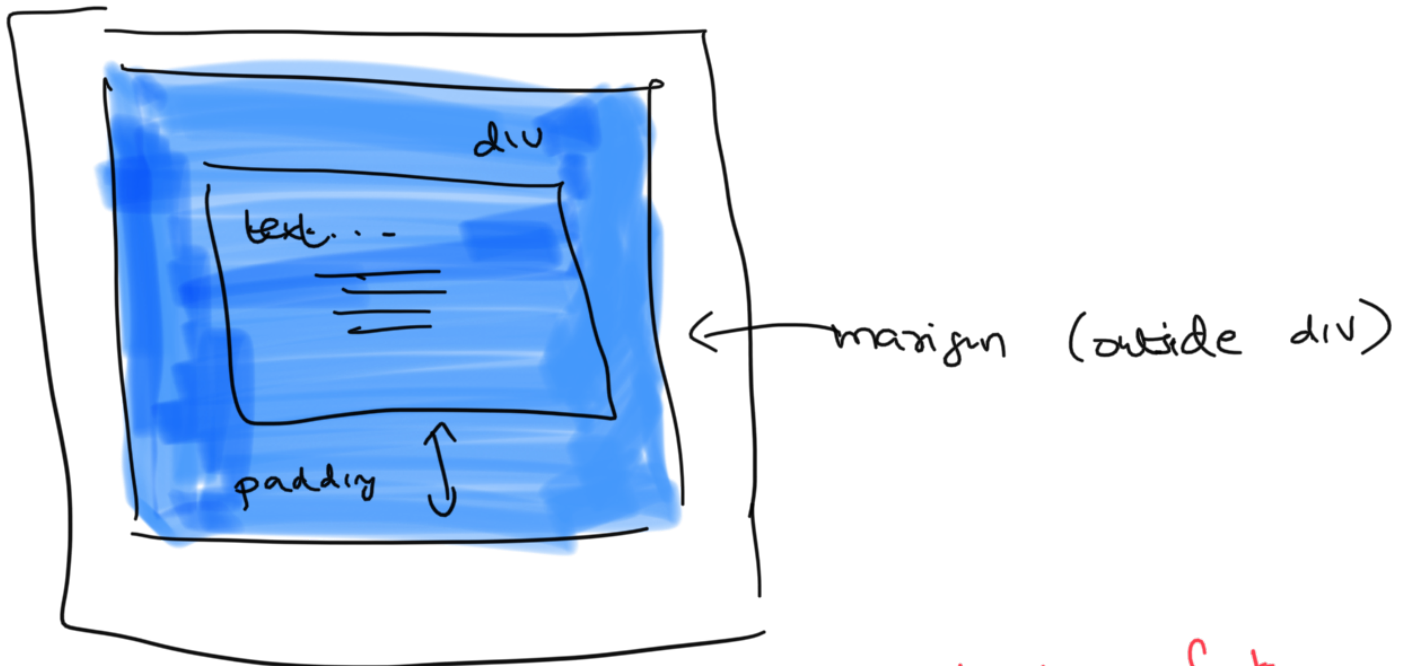
background-color: blue,

width: 100px; ← px is pixel. etc..

margin: 20px; — "invisible"

margin to your element"

padding : 20 px;



back up font  
in case first choice  
not available

font-family : Arial , sans-serif ;

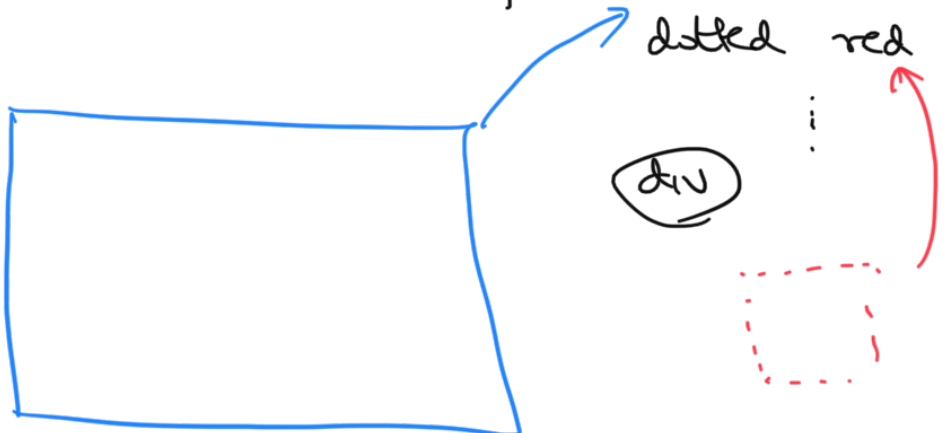
font-size : 20 px;

font-weight : bold ;

## Borders

Div ← usually invisible box

can say border : 3px solid blue ;  
dotted red



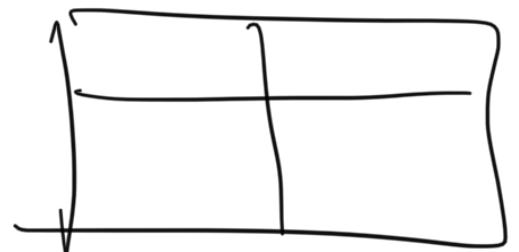
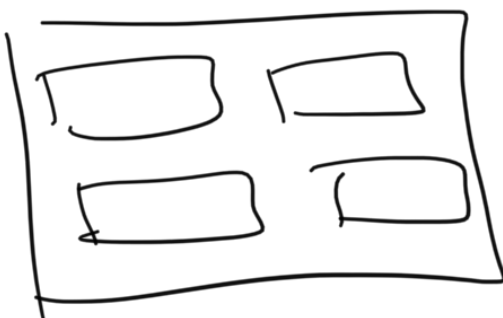
can say

```
td, th {
  properties
}
```

← same style properties applied to `<td>` `</td>` `<th>` `</th>`

table

→ `border-collapse: collapse;`



can also say

```
th
td {
  properties
}
```

```
th {
  additional properties.
}
```

styling using ids / classes

# my-id

{  
these  
properties

}

←  
applied to  
element with  
id = my-id

• my-class

{  
||||

}

←  
applies to  
all elements  
with class  
"my-class".

## CSS selectors

① styling children and descendants

<ol>

<li> 1 </li>

<ul>

<li> 2 </li>

</ul>

<li> 3 </li>

</ol>

<ul>

<li> 4 </li>

</ul>

output

1.	1
2	0 2
3	.3
•	4

② If in CSS file, you say

ol li {

} any li  
which is

descendant selector  
 $a > b$

space

```
{  
  color: red;  
}
```

= descendant  
of ol  
gets coloured  
red

⑥ If in CSS file, you say

child selector  
 $a > b$

```
ol > li {  
  color: red;  
}
```

any li  
which is  
= direct  
child of  
ol

⑦ adjacent  
sibling selector  
is  $a + b$

1. 1  
2. 02  
3. 3  
• 4

② styling differently attributed tags...

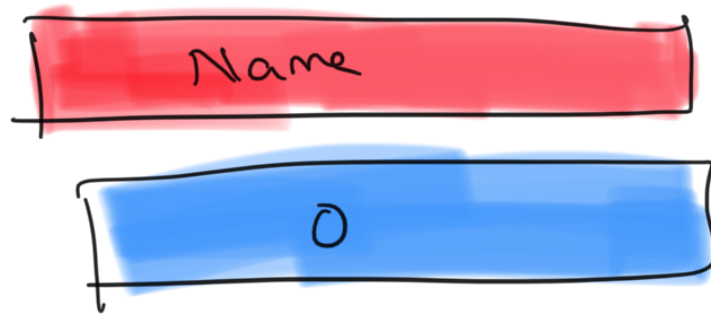
```
<input name="name" type="text"  
        placeholder="Name">
```

```
<input name="age" type="number"  
        placeholder="0">
```

In CSS file, can say

```
input [ type = "text" ] {  
    background-color: red;  
}
```

```
input [ type = "number" ] {  
    background-color: blue;  
}
```



### ③ pseudo-class

HTML

```
<button> click!  
</button>
```

CSS

```
button {  
    width: 200px;  
    height: 50px;  
    font-size: 24px;  
    background-color: green;  
}
```

*pseudo class -  
rep state of  
html element -  
button*

```
button: hover {  
    background-color: blue;  
}
```

click

click



click



when user hovers  
over button,  
background is  
blue

#### ④ pseudo-elements

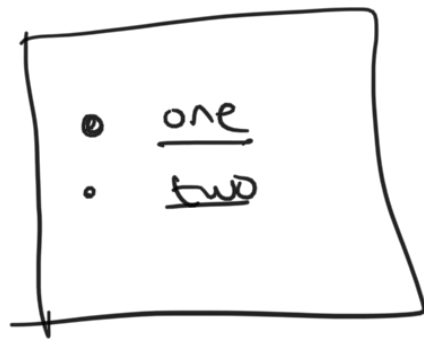
- ways of affecting particular parts  
of particular html element

<ul>

<li> <a href="..."> one </a> </li>

<li> <a href="..."> two </a> </li>

</ul>



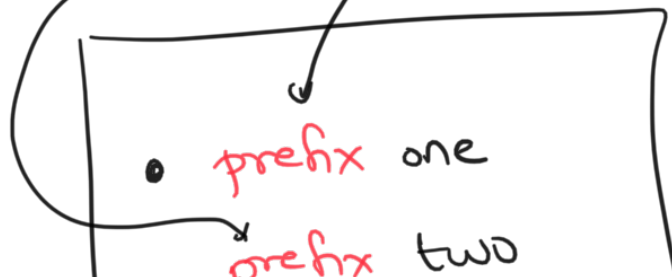
if CSS

a :: before {

content : "prefix";  
font-color : "red";

}

adds this  
content  
before the  
element's  
actual  
contents







stray info :

⇒

↔

"\21d2"

[hex value  
for ⇒  
unicode  
symbol]

p::selection {  
...  
}



styling applied  
to selected  
part of  
paragraph  
--

<p> blah blah blah </p>

Isn't this  
pseudo class  
though,  
why is it  
pseudo element?