JAVASCRIPT

Date
DELTA Pg No.

(MERN)

- MongoDB → Database
- Express → Backend
- React → Frontend libraries
- Node.js → Frontend

→ Installing NodeJS
  ↳ 16.13.1 LTS version
  ↳ Open CMD
  ↳ node --version (To see the nodejs version)
  ↳ node (To open node)

→ Vs code
  ↳ open folder
  ↳ Terminal
  ↳ New terminal

→ First javascript code (ex)
  ↳ Create a file in folder (test.js)

code

1. console.log('Hello Rahul')
   ↳ To run → New terminal → node test.js

output → Hello Rahul

→ create index.html
↳ type → ! and press tab

SKELETON OF WEBSITE

- index.html

```
<!DOCTYPE html>
<html lang="en">
Start ——→ <head>

end ——→ </head>
start ⟹ <body>

end ⟹ </body>
      </html>
```

→ Install live server
↳ Ritwick Dey

⟹ **Variables**
↳ no need to define variable type.
↳ JS is dynamically typed variable

```
var a = 2
```
↳ Keyword

```
var a = 'I am a string';
var a = true
console.log(a)
```

output

- Keyword

1. Var → variable can be redeclared.
2. let → variable can be reassigned but not redeclared.
3. const → no redeclaration, no reassignment

⟹ JS treats function as first class citizen
↳ JS doesn't differentiate between fn and variable.

⟹ IIFE (Immediately invoked function Expression)

example

```
1. let add = (function(a,b){
2.             return a+b
3.           })(10,20)
```

↳ call

Install code runner for easily running the code.

ARRAYS

- Arrays in JS can contain element of different datatypes in same array (unlike JAVA and other lang.)
- Array declaration
  ↳ let arr = [ ]
  ↳ let arr1 = ['Rahul', 2, false, null]

⟹ Array Methods

(1) Pop method( ) → Removes the last element of the array (Inbuilt method)

→ arr.pop( )

(2) push() method → Adds the element at the end of array.
→ arr.push('value')

(3) shift method → removes the first element of the array
→ arr.shift()

(4) unshift method → adds the first element of the array.

→ Multidimentional Array

```
let matrix = [
    [1, 2, 3],
    [9, 5, 6],
    [7, 8, 9],
]
```

STRING

→ It is a sequence of characters

Methods

(1) → Extracting a part of string
(a) slice(start, End+1)

```
let str = 'Rahul Anand'
str.slice(0,5)
```

output → Rahul

(b) substr (start , length)
str.substr (5, 5)
Output → Anand

⇒ Replacing string content

* str.replace ('007', '7712')
output → Rahul Anand 7/2

⇒ toUpperCase() and toLowerCase()

Capital — Small

str.toUpperCase — str.toLowerCase

⇒ Concatenation (concat Method)

let firstString = 'Hello'
let secondString = 'Rahul'
consol.log (firstString.concat(" ", secondString)

Conscat method

⇒ Trim method → used to remove white spaces (extra)

str.text.trim()

OBJECTS

- let object = {

}

- let person = {
  name = 'Rahul',
  age = 23,
  phNo. = 12345,
  friends = ['Vivek', 'Manav']
}

key value

key

you can store array in object.

→ We can also store functions & objects inside an object.

for in loop (objects ke liye)

→

```
for (let key in person)
    console.log("key:", key, "value", person[key])
```

HTML

HTML → Hyper text markup language.

⟹ SKELETON

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta ...>
    <meta ...>
    <meta ...>
    <title> MY Website </title>
</head>
<body>

</body>
</html>
```

Head start → `<head>`

`<meta ...>`, `<meta ...>`, `<meta ...>` } no changes

`<title> MY Website </title>` → title of webpage

Head stop → `</head>`

`<body>` → body start

`</body>` → body end

`</html>` → code end

- To comment in HTML

  `<!-- comment -->`

  start: `<!--` end: `-->`

⟹ HEADINGS (Inside the body)

```
<body>
    <h1>
        I am Heading 1
    </h1>
</body>
```

- h1, h2 .. h3 .. h6 → can go upto h6

as no. increases, size of font decreases

⟹ PARAGRAPHS

 Paragraphs can be written
start end

⟹ DIVISION

 → start

 → end

⟹ LISTS

↳ UNORDERED LIST

```
<ul>
    <li> Item A </li>
    <li> Item B </li>
</ul>
```

↳ ORDERED LIST

```
<ol>
    <li> Item D </li>
    <li> Item E </li>
</ol>
```

⟹ SPAN TAG

- ...

→ To change properties

- ...

⟹ Anchor Tag (for parting link)

 text

link to this site

⟹ IMAGE TAG

property

⟹ CSS SELECTORS #IMP

↳ used for web scraping

used to get important or required details

- css is typed inside style tag.

Selectors (css)

→ (1) element selector

```
<style>
    h1 {
        color : Blue;
    }
</style>
```

Always use ; (semicolumn) in css.

→ (2) Decendant selector

```
<style>
    div a {
        color: Blue;
    }
</style>
```

→ ③ Children selector

only the next tag can be selected in line (can't skip tag)

```
<style>
div > h1 > span {
    color: Blue;
}
</style>
```

used for this selector

→ ④ class selector

```
<style>
.class1 {
    color: Blue;
}
</style>
```

Dot is used

color of these 2 will change

① `<p class = "class1"> . . . . </p>`

② `<p class = "class1"> . . . . </p>`

③ `<p class = "class2"> . . . . </p>`

→ ⑤ Multiple class selector

```
<style>
.class1.class2 {
    color: Blue;
}
</style>
```

colour of all three will change.

→ ⑥ Descendent class

```
<div class="c1">
<div>
    <div>
        <span>
            <div class="c2"> line 1</div>
        </span>
    </div>
</div>
```

change colors of this line →

```
<style>
    .c1 span .c2{
        color: Blue;
    }
</style>
```

→ ⑦ Class with children combinator

```
<style>
    .c1 > p {
        color: Blue;
    }
</style>
```

• for children

Date
DELTA Pg No.

→ (8) ID Selector

```
<p class = "s1"> Para 1 </p>
<p class = "s1" id = "theOne"> Para 2 </p>
```

To change color →

```
<style>
#theOne {
    color : blue;
}
</style>
```

- hash tag for ID selection

→ (9) Attribute Selector

```
<form>
    <input type = "button" value = "Click Me">
</form>
```

change color →

```
<style>
input [value = 'click me'] {
    color : Blue;
}
```

- Attribute selector