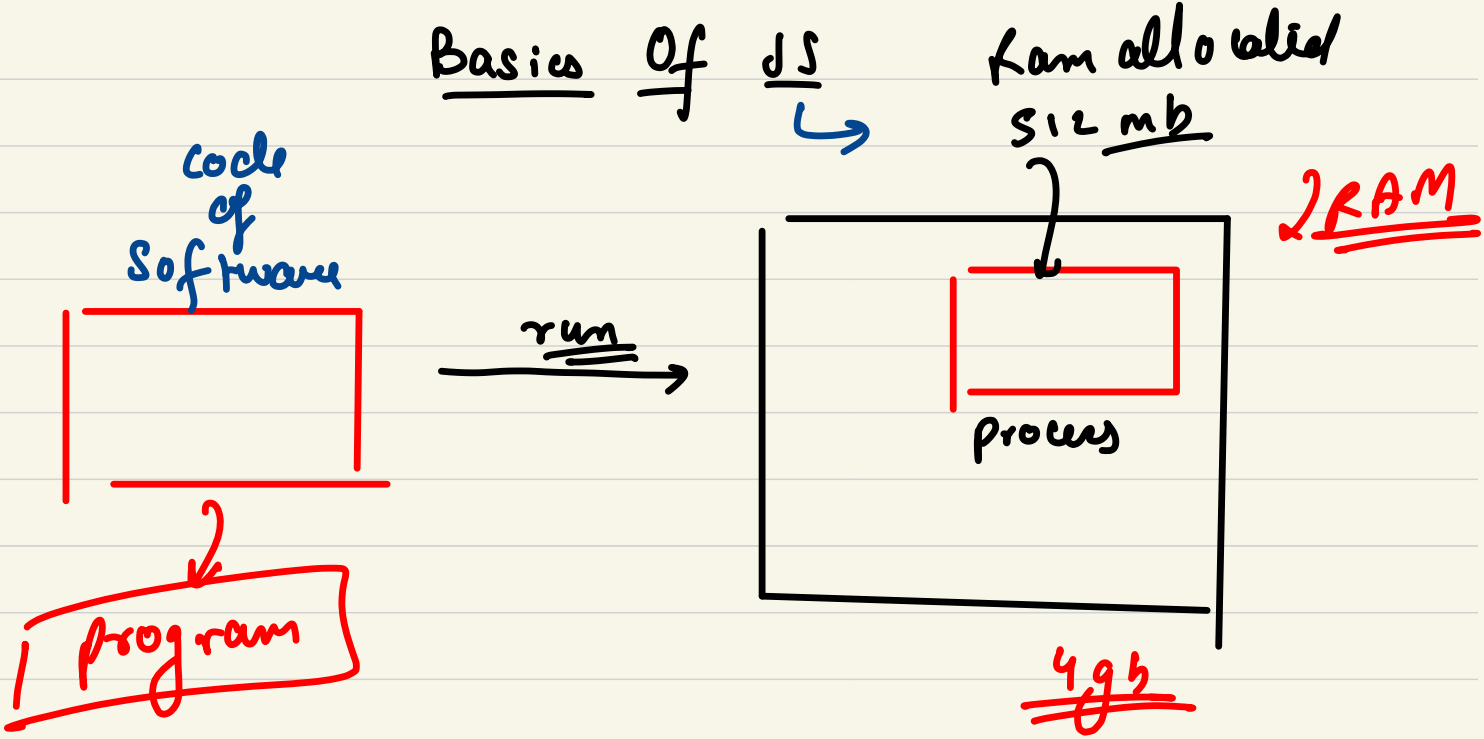
Basics Of DS



Program in a Running state is called process.

Keywords

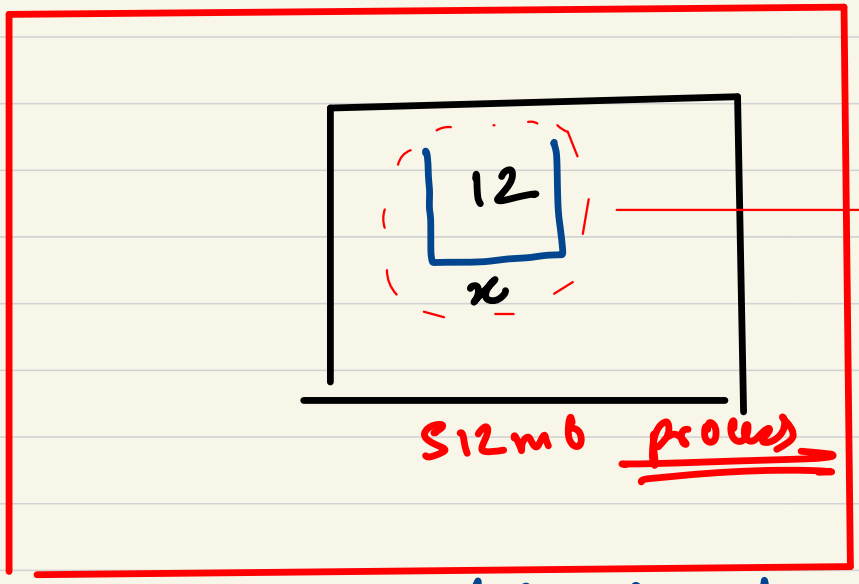
↳ In a programming lang, few words are reserved by the lang for its own purpose. We cannot change these meaning or repurpose them for a different use case.

for eg → In JS few keywords are -

let, for, const, if, else, return etc.

How can we store some values inside a program

→ To store values we can use **Variables**.



Variable are memory buckets that store our values & has a name as well.

Javascriptfile →

.js extension

index.js

demo.js

↳ open vs code

↳ click on file → click on open folder

↳ In the popup create new folder

↳ Once you created a folder open it.

↳ Click on "new file" icon inside vs code.

↳ In the dropdown name your file & press enter.

↳ How to create variable in JS ??

3 ways to create variables →

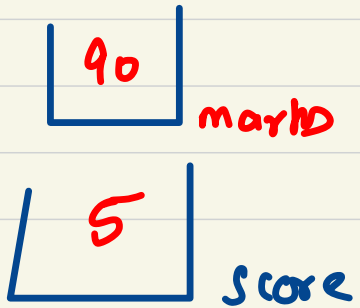
1) using var Keyword

→ var .variablename = value ;

forever

var marks = 90;

var score = 5;

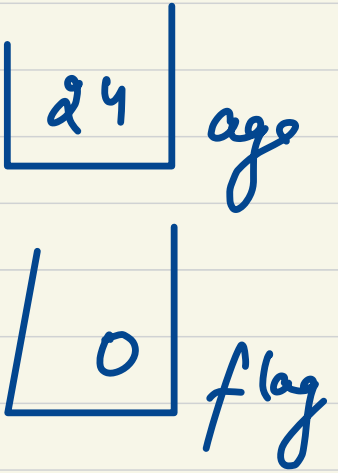


2) using let keyword

Syntax → let .variable name = .value ;

Ex → let age = 24;

let flag = 0;

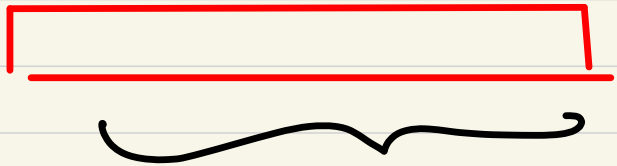


3) using Const keyword

Const variablename = value;

Ex Const x = 100;

Semicolon



one line of JS code

Equal to one instruction that we want to give to the computer.

One complete instruction is called **Statement**.

At the end of every statement in JS we can put a semicolon (but it is optional)

Naming Rules for variable creation

1) Variable can contain small alphabets, capital alphabets, digits, underscore (_), dollar ($)

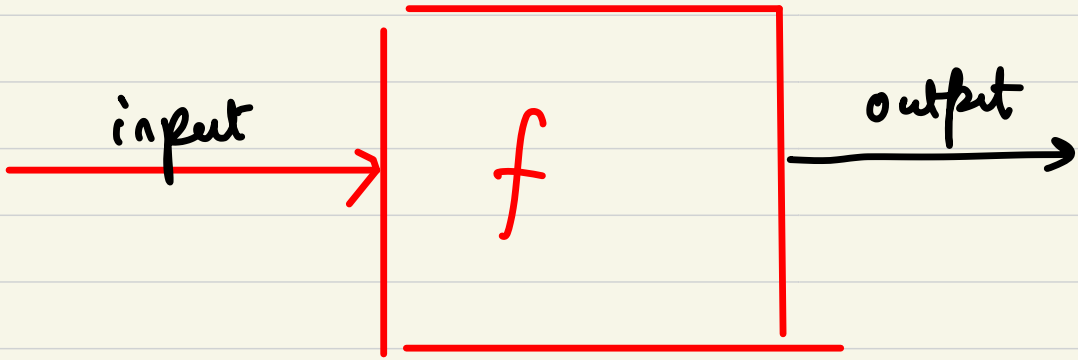
2) We can't have space or any other special char.

3) We should give meaningful names so that reader can identify the names properly.

4) variable name cannot start with a digit but it can have digits in between or at last.

Ex let 0james = 10; ✗ wrong syntax

5) We cannot use keywords as variable names.



functions
as a
Black box

$f(x, y \ldots) \longrightarrow$ =

1

console.log ( )

this function takes some data as input and then displays them in the output

→ if we have 2 different console.log() , then both of them displays / prints output in different lines.

we can also pass multiple values together to console.log , & all of them will be on same line.

```
let age=23;
console.log(10, "starting", age);
```

What all values we can store in JS ??

(1) Number → 10, -3, 2.6, 3.444, 100, 10000 etc

var x=10;

(2) String → text

" Sanket (") → double quotes

' Random Text (') → single quotes

` Some text (`) → backtick

var name = "Sanket";

3) boolean → true

0/1

false

} keywords

→ these are all small values.

```
var condition = true;
```

4) undefined → keyword

→ Something not defined yet but may be defined later.

```
var status = undefined;
```

5) objects → If we have to somehow store key-value pairs, then we can use objects

<key, value>

user1 =
{
name: "Sanket"
company: "Google"
position: "Software Engineer"
}

user2 =
{
name: "Sarthak"
company: "Phonepe"
position: "AI/ML"
}

Keys will be unique.

```
{ name: "Iphone",
  cost: 130,000,
  color: "deep purple",
  discount: 5,
  description: "the new apple Iphone",
}
```

Primitive

2

types which are atomic in nature

Eg → number

Non primitive

2

types which are a composition of other types.

Eg → object

user = {

name: "Sanket"

age: 24

posts: {

createdAt: "Jun 12, 2022"

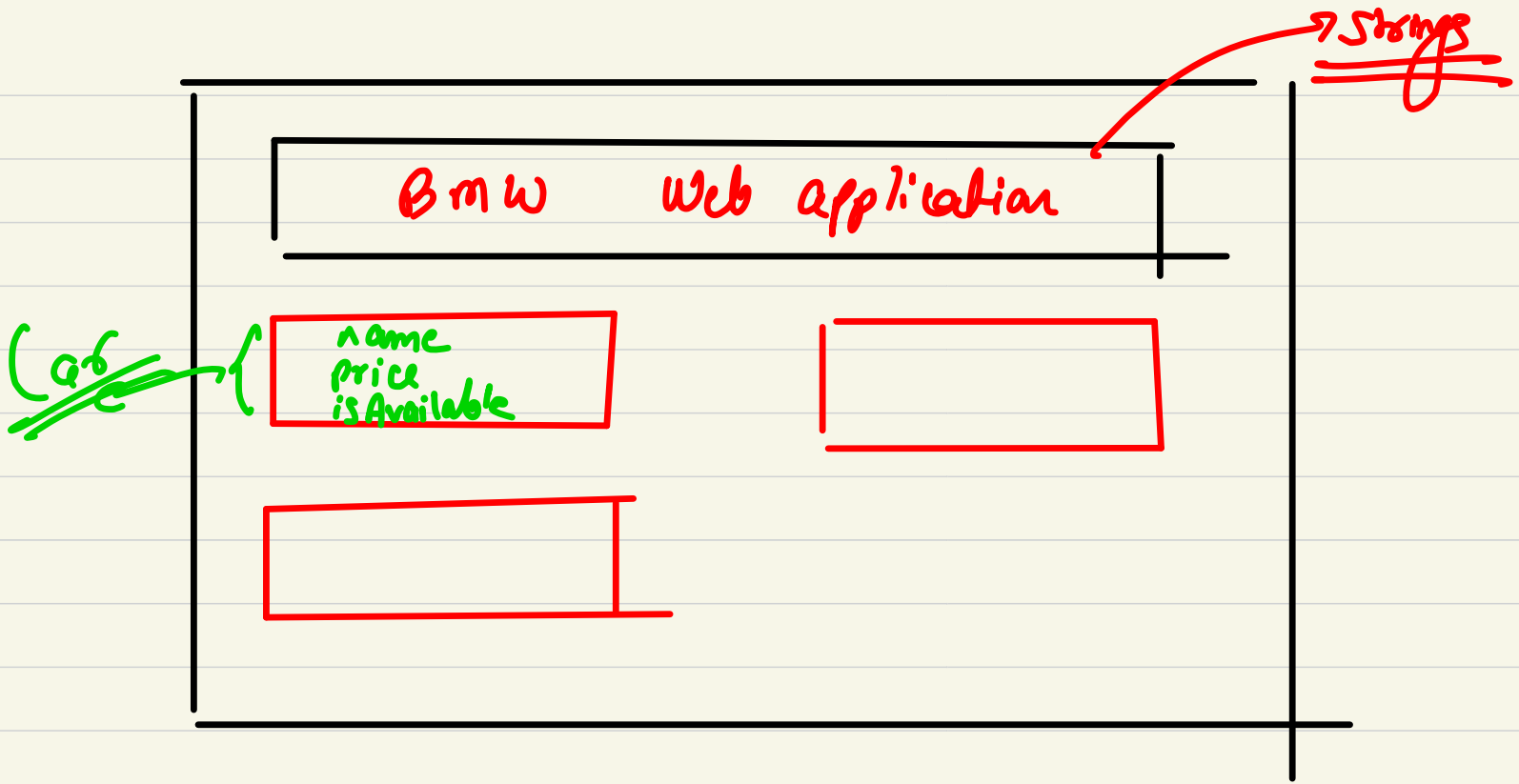
text: "My first post"

}

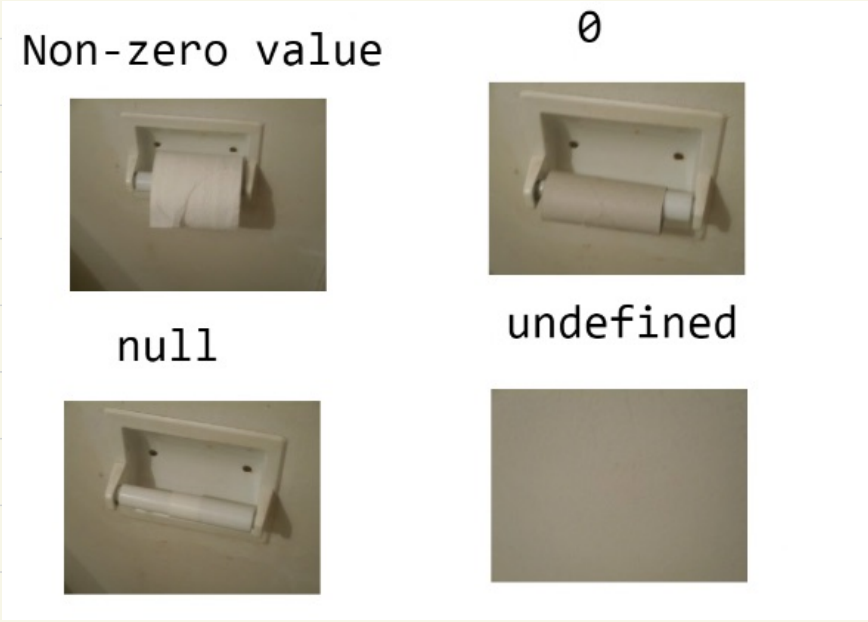
gender: "Male"

...

}


Strings
BMW Web application
name
price
isAvailable

6) Null → It actually represents empty value.



let a;
↳ undefined

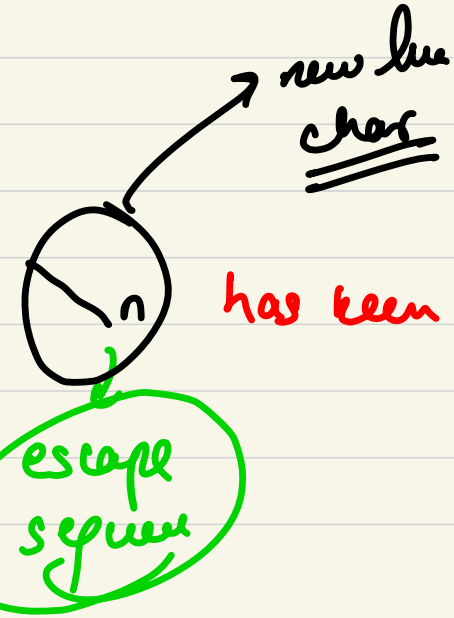
let b=10;
b=null;

Special chars

\t → tab

"The new apple 'phone"

"has been launched"



"The new apple Iphone has been launched"

: → colon

; → semicolon

Comments

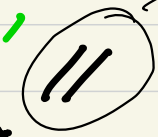
↳ Comments are a piece of code that is for documentation purpose.

↳ the programming lang, will just avoid comments during execution.

→ In JS, there are 2 ways,

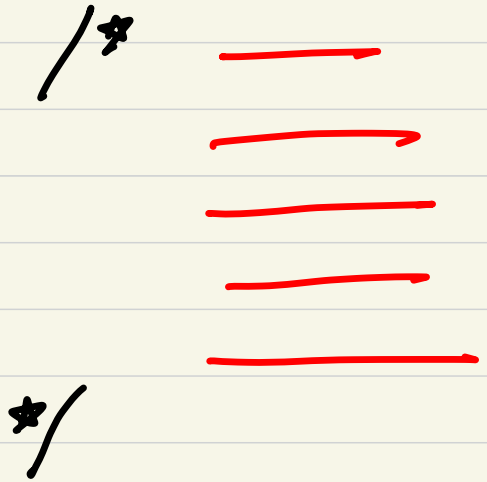
→ Single line comment → //

→ Multi line Comment → /* ... */



// This is a comment

or



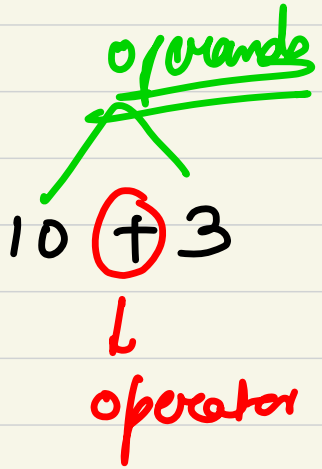
Operators

operand → values on which we want to do the operation

There are different types of operators →

① Arithmetic Operators:

+	→	addition
−	→	subtraction
/	→	division
*	→	multiplication
%	→	remainder
**	→	exponent



a^b → a**b

Assignment Operators

=, +=, -=, *=, /=, %=

let a = 10 ; → assignment operator.

→ = just directly assigns the value on RHS to LHS.

a += 2 → a += 2 means → a = a + 2

let's say prev value of a was 10, then

if you do a += 2, → a = a + 2

on LHS we take old value of a & add 2 to it

a = 10

a += 2 ⟶ a = a + 2

Take prev value of a, add 2 to it, then assign this new value back to a.

if prev value of a was 10, then after this operation it becomes 12.

a -= 2 ⟶ a = a - 2

a *= 2 ⟶ a = a * 2

a /= 3 ⟶ a = a / 3

a %= 3 ⟶ a = a % 3

var a = 10

var a = 2

let a = 10;
let a = 10; ✗ ⟶ error