__/__/____

UNIT-3 "JAVA SCRIPT".

1. It is used to create dynamic webpages.
2. JS can be written with or in an HTML webpage and will run automatically when page loads. It is executed in plain Text.
3. JS can be executed in browser / server and a special program on device Javascript Engine.
4. The embedded engine is known as JS Virtual Machine.

Different Engine in Different Browser.

1. V8 → Chrome / Edge / Opera
2. SpiderMonkey → Firefox
3. Chakra → IE (Internet Explorer)
4. JS core, NITRO, SQUIRRELFISH → Safari

Characteristics →

1. It is safe because it doesn't provides low level Memory Access/CPU.
2. It supports func^c allow that JS to read / write arbitrary files, perform network requests
3. It can manipulate a webpage.
4. It can interact with user Ex mouse, get and set cookies, show messages.
5. It remembers data on client side.

Restrictions →

1. Due to high flexibility JS is restricted
2. It can't read / write arbitary files on hard disk, copy them or execute program. It has no direct access to OS function.
3. For these functions it requires explicit permission.

Languages related to JS →

Coffeescript, Typescript, Kotlin, Brython, DART, FLOW

Absolute Path → <script src = " js/script/java.js "></script>

Relative Path → <script src = " script.js "></script>

<SCRIPT> TAG →

```
<!DOCTYPE html>
<html>
    <body>
    <script type = "text/javascript">
      alert ("HelloWorld");
    </script>
```

Concept of semicolon

```
<script type = "text/javascript">
alert ("Hello");
alert ("Hello again");
```

If we remove semicolon then also o/p will be displayed as it is known automatic semicolon insertion in javascript.

Q
```
alert ( 3
        +1
        +2 );
```

Ans) Semicolon insertion fail.

Semicolon Insertion fail →

Q.
```
alert ("Hello")
[1,2].forEach( alert );
```

→ Hello 1 2 § Sol"

→ Semicolon Insertion Fail

/ /

Variable →

→ var or let

→
```
let dhinchak;
dhinchak = pooja;
alert (dhinchak)
```

o/p → Pooja

```
let dhinchak;
dhinchak = pooja;
dhinchak = "Anshika";
alert (dhinchak)
```

o/p → Anshika

Naming Convention of a Variable

→ 2 limitations on variable name in JS

1. The name must contain only letters, digits or symbols ($, _____)
2. The first letter must not be a digit.

eg - usertext2 ✓

youarexc ✗

```
let $ = 1;
```

* **Datatypes** → There are 8 basic datatypes in JS.

↳ We can put any type in a variable

Eg A variable can add a string as well as store a number.

let apple = "How are you?"

↳ The programming lang which allow such thing such as JS are dynamically right, meaning that there exist datatype but variable are not bound to any of them.

① **Number**: int or float

↳ There are also special numeric value associated with this datatype

→ infinity • alert(1/0) → ∞
• alert(infinity) → ∞

→ NaN represents Computational Error.

Ex:- if we say alert("Not a Number"/3)

Nan is sticky

→ −∞

② **BigInt** → Number type can't safely represent Int values larger than $2^{53}-1$ or less than $-2^{53}-1$. Outside safe Integer Range +, $-2^{53}-1$ there will be a precision error because not all digits fits into fixed 64 bits storage.

③ **String**

④ **Boolean** : True or False

⑤ **The null value** : not a datatype. It forms a seperate type of itself. It means empty, nothing or unknown.

⑥ **Undefined Value** → undefined is when value is unassigned

⑦ **Object**: → This type is a special, used to store collection of data & more complex entities. All other types are called (primitive) because there value contain only a single thing

__/__/____

★ Symbols → used to create unique Identifiers for object...

★ type of operator → type of (NAN) → undefined
type of (Math) → object
type of (NULL) → object
type of (alert) → function

INTERACTION IN JS →

1. ALERT → OK Button 1 interaction button
2. PROMPT → prompt (title, [Default]); "OK" & "cancel" (2)

eg
```
let age = prompt ("What is ur age", 18);
alert (age);
```

3. CONFIRM → result = confirm (question)

→
```
let n = 5 > 4;
confirm ();   True / False...
```

TYPE CONVERSION

1) STRING → let weatherupdate = true; → Boolean.
String (weatherupdate); → String

2) NUMERIC → let n = "1, 2, 3 4 5"; → String
number (n); → numeric

eg:
```
alert ("1" + "2") → 12
alert ('1' + 2) → 12
alert ("6" / "2") → 3
```
} Only works with '+'

RULES

1)
```
let adam
alert (Number (adam)); // NaN (Not a numb
alert (Number ("123")); // 123
alert (Number ("123z")); // NaN
alert (Number ("True / False")); // 1/0
```
can't convert to num.

__/__/____

BOOLEAN RULES

1) Values which are empty like '0', NULL, empty string, undefined and NaN becomes false.
2) All other values are "true".

eg:- alert (Boolean ("0")) // True

Exponential → "**".

String Concatenation with Binary +

alert ('1' + 2 + 2) → 122

Converts to string....

Numeric Conversion with Unary +

let square = "15";
let rect = "7";
→ alert (square + Rect); // 157
→ alert (+square + +Rect); // 22

Unary plus converts string to number. It is equivalent to Number();

Operator Precedence

1. Unary +
2. Unary −
3. Exponential
4. Multiplication
5. Division
6. Addition
7. Subtraction
8. Assignment

let age = NULL ✓ not undefined
let age ; undefined

__/__/____

1. '=' → Assignment Operator
2. '==' → Conditional Statement
3. '===' → Strict Equalizer

→ (NULL == undefined) → true
→ (NULL === undefined) → false (We can't say NULL ≠ undefined)

✓ alert (NULL == 0) → false
✓ alert (NULL >= 0) → true

alert (undefined > 0) → error.
alert (undefined == 0) → error
alert (undefined >= 0) → error

} as it is incomparable We can't compa