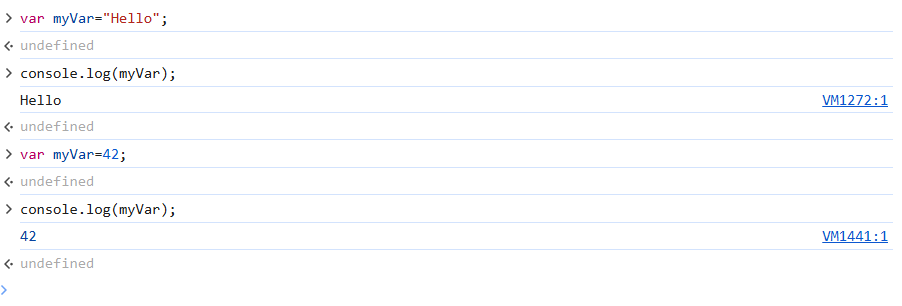
Java Script Assignment-1

1.

var myVar="Hello"; //String

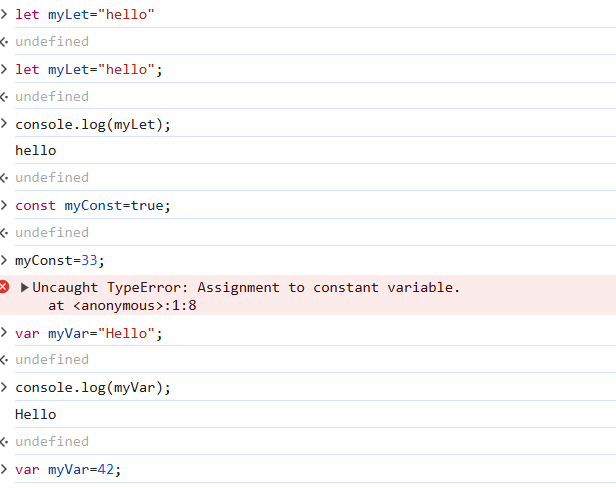
let myLet=42; //number

const myConst=true;//Boolean

console.log("Var:", myVar);//Hello

console.log("myLet: ",myLet);//42

console.log("myConst:",myConst);//a Boolean



myVar=42;

myLet=true;

console.log("myVar:", myVar);//42

console.log("myLet:",myLet);//true

var:

it can be reassigned and redeclared

let:

it can be re-assigned , but not re-declared.

const:

it cannot be re0-assigned and nor re-declared.

2



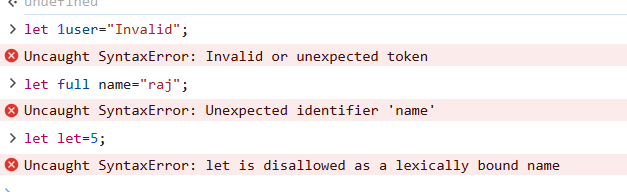
5 valid Variable names:

These are valid because:

->They start with a letter, undersore\_, or dollar sign $

-> No saces or special characters

-> Not using reserved keywords



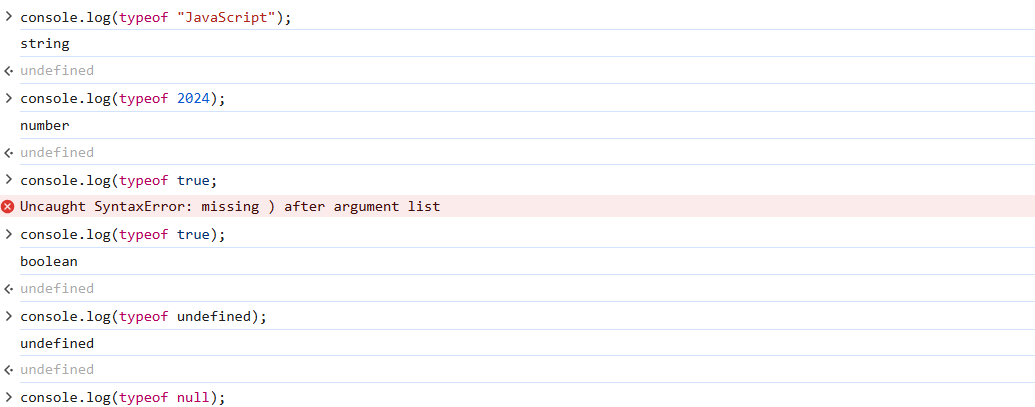
3 Invalid Names:

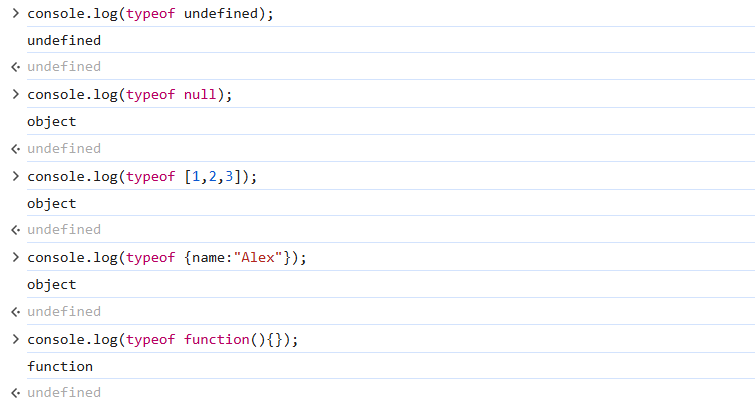
-> Variable names can't start with number.

->variable names can't contain spaces.

-> "let" is a reserved keyword in JavaScript.

3.





Type of Operator:

console.log(typeof "JavaScript");

string

//String- A sequence of Characters

console.log(typeof 2024);

number

//number- Numeric Value

console.log(typeof true);

Boolean

//boolean- true or false

console.log(typeof undefined);

undefined

//undefined - variable declared but not assigned

console.log(typeof null);

object

//Null- null is not actually an object , but returns "object" for backward compatibility

console.log(typeof [1,2,3]);

object

// use Array.isArray([1,2,3]) to properly check for arrays

console.log(typeof {name:"Alex"});

object

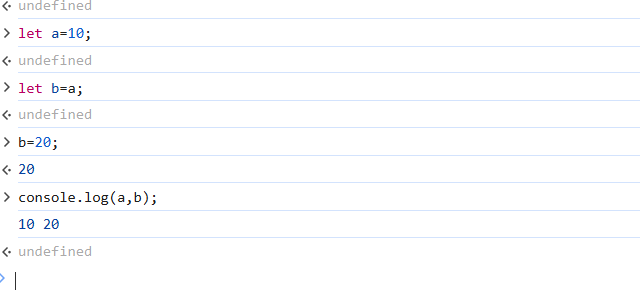
//object- standard JavaScript objects

console.log(typeof function(){});

function

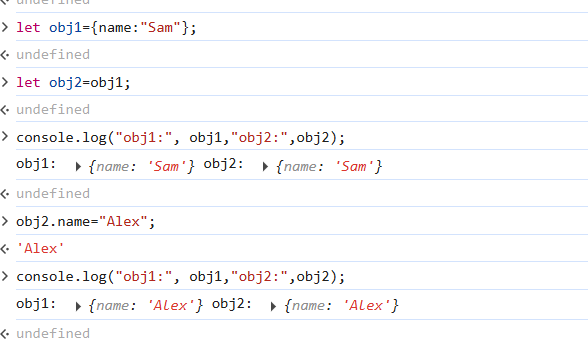
//function - A callable object

4.



condition let a=10;

the value of the variable initialized to '10' and when we created 'b' variable assigned to 'a' the value of 'b' is also '10' and then we assigned new value '20' and the assigned value of 'a' to 'b' got updated to new assigned value and 'a' is not affected because it is primitive type. output of 'a' and 'b' is '10', '20'.



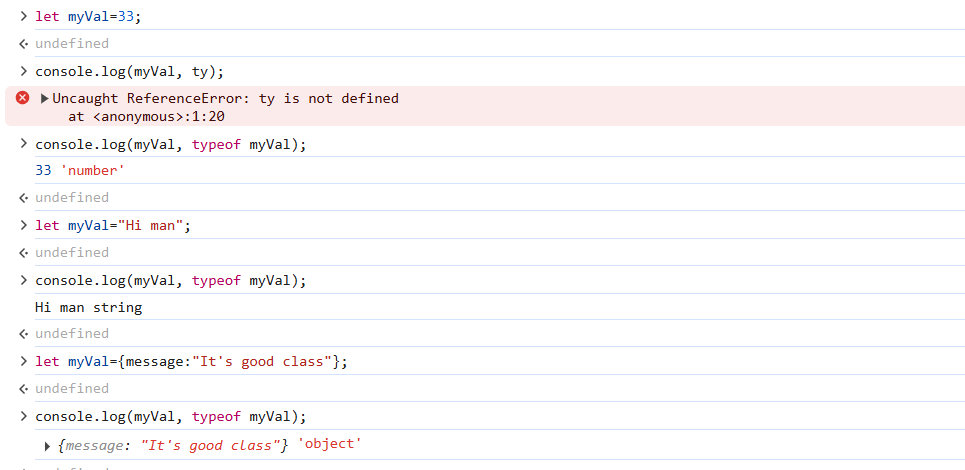
obj1 holds a reference to an object in memory.

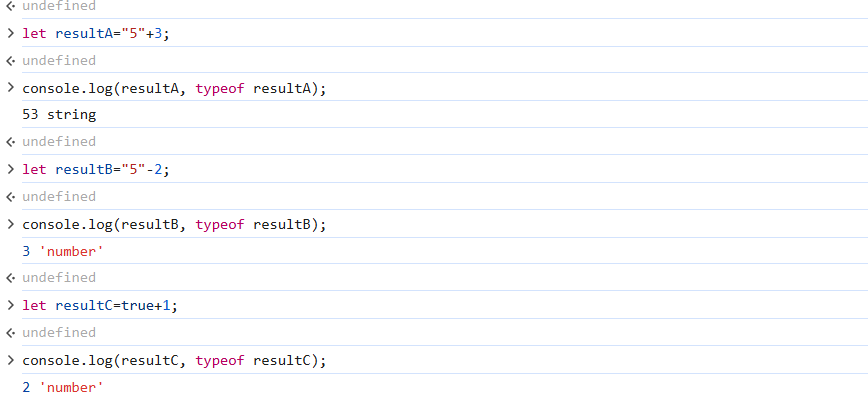
obj2 = obj1 makes obj2 point to the same object.

Changing obj2.name also changes obj1.name.

Objects (including arrays and functions) are copied by reference.

5.





A)"5" is a string, and + is string concatenation when one operand is a string.

So, 3 is coerced to "3" and concatenated → "5" + "3" = "53"

Final Type: string

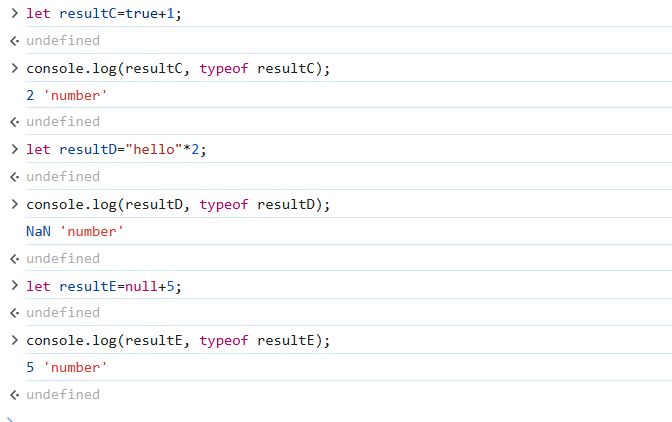
B)- forces numeric context.

"5" is coerced to number 5, then 5 - 2 = 3

Final Type: number

C)true is coerced to 1, so 1 + 1 = 2

Final Type: number



D)"hello" can't be converted to a number.

"hello" \* 2 results in NaN (Not a Number)

NaN is still a number type

Final Type: number

E)null is coerced to 0, so 0 + 5 = 5

Final Type: number