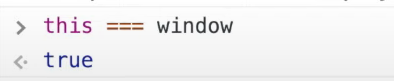
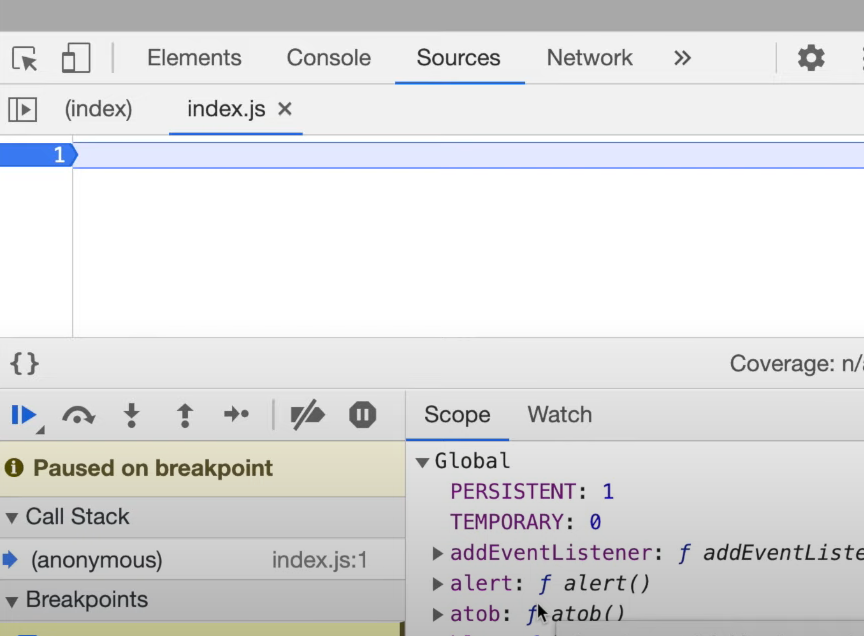
### Window & This

Window is a global object which is created along with the execution context . Whenever any javascript code runs global object is created , global execution context is created and along with the execution context ‘this’ variable is created.

Javascript will run on browsers, servers etc.Every browser has its own javascript engine. Chrome has v8 engine, safari has its own, firefox, microsoft have its own js engine. All these js engines have responsibility to create this global object . In case of browsers it is known as window, in case of node it would be something else. Even the js file is empty js engine will create this global object,GEC and call stack. This will be created for both functional and global execution context

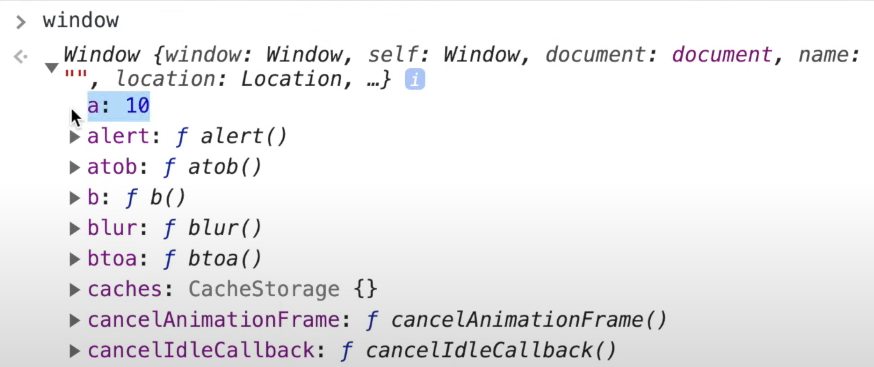
At global level, this === window in case of browsers

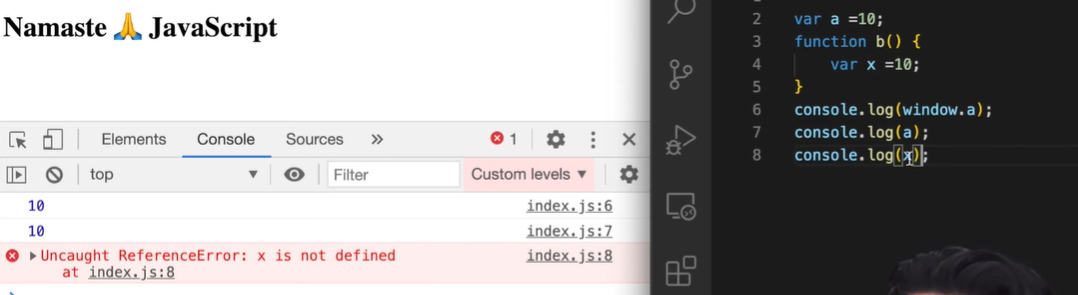


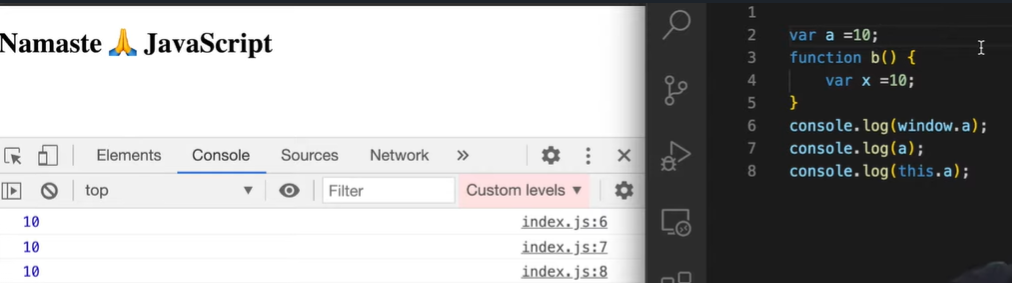


| **Global Space**  var a = 10;  Function b()   | **Local Space**  **{**  var x = 10;  **}** | | --- | |
| --- | --- |

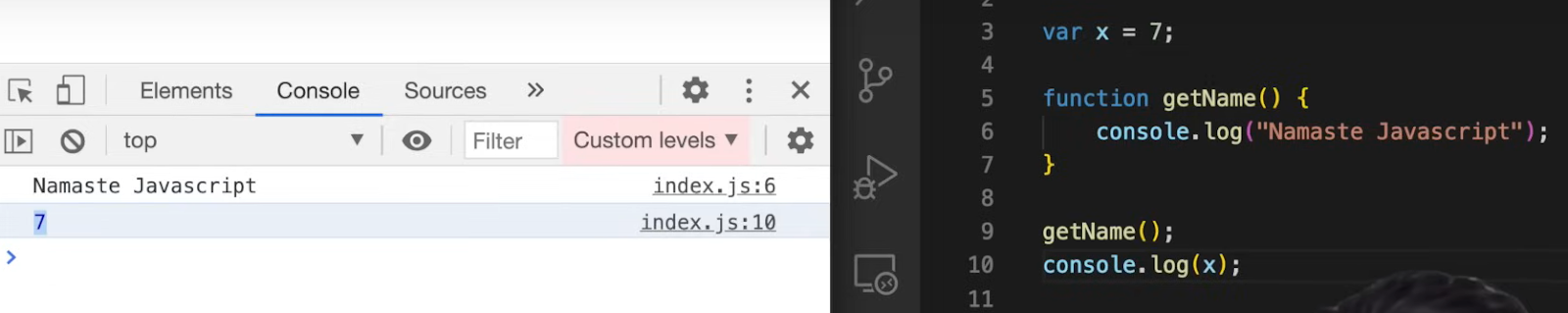
**The variables and functions are attached to the global object window**

****

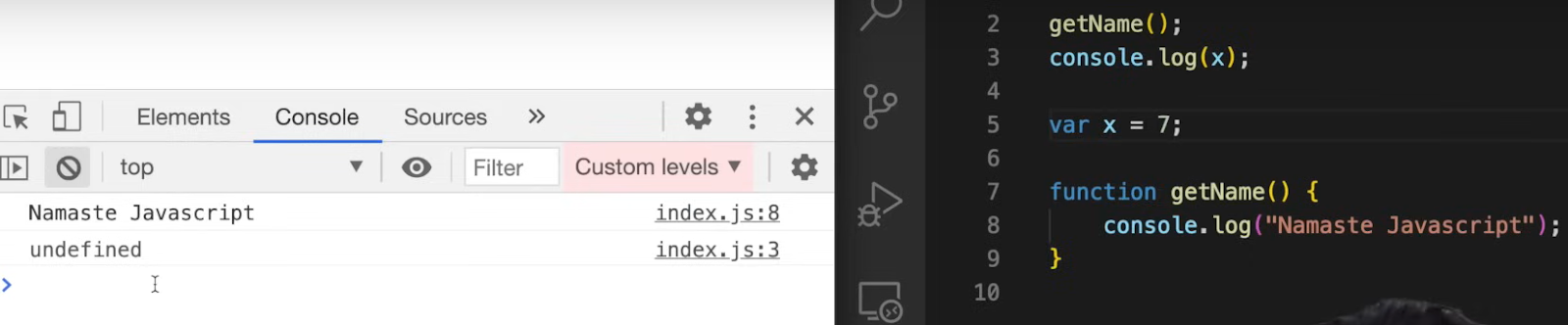
****

****

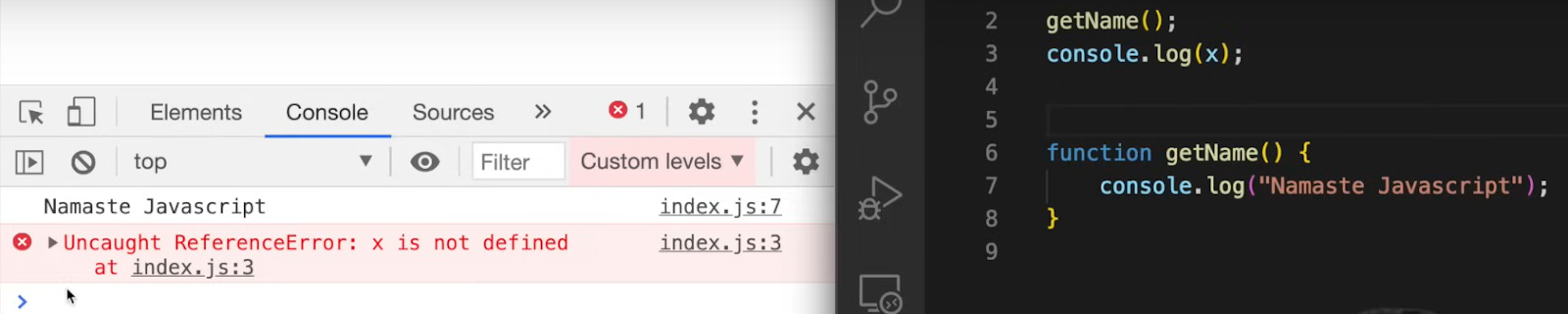
### Hoisting in Javascript



**Calling function and variable before declaration**

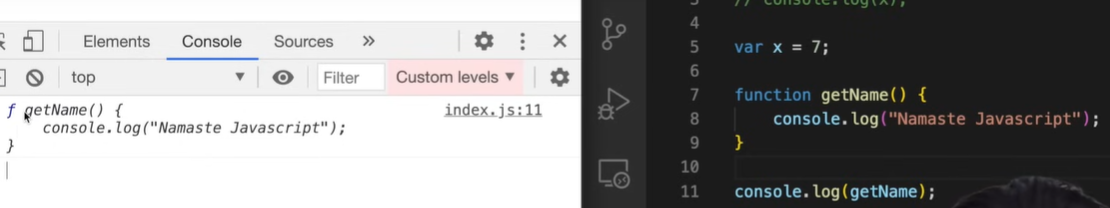
****

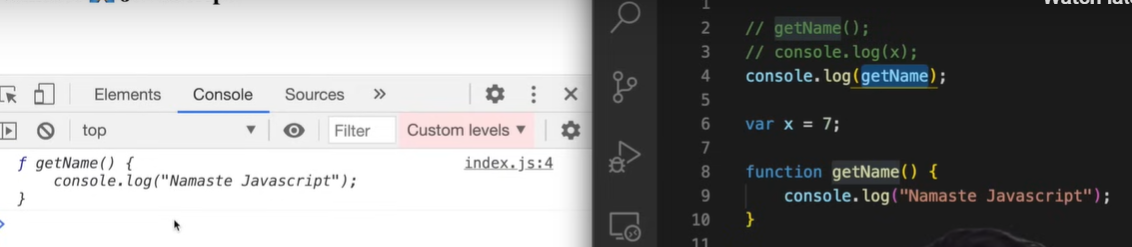
It would give error in other languages but not in javascript. U can call function and access the variable before its declaration



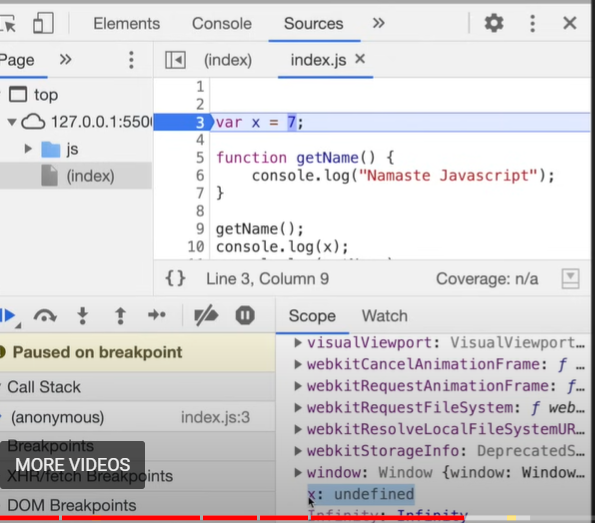
#### Hoisting

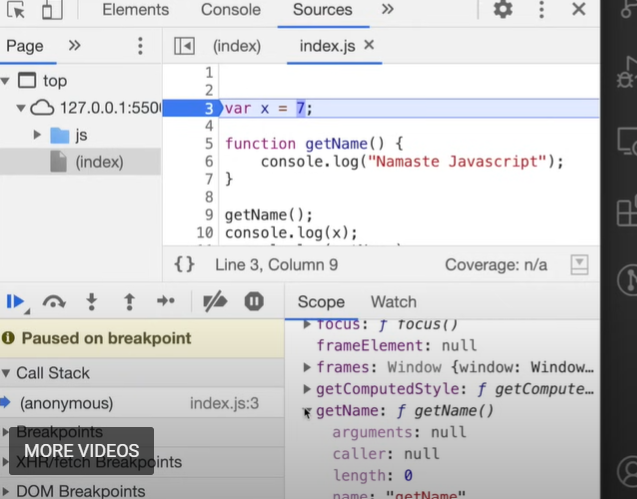
is a phenomenon in Javascript by which you can access these variables and functions even before u r initializing it. In Javascript before code executes Global execution context will get created and in memory phase itself variables with value undefined and wholefunction code will be copied in memory. So we can access variables and functions before its declaration. This is called hoisting



****

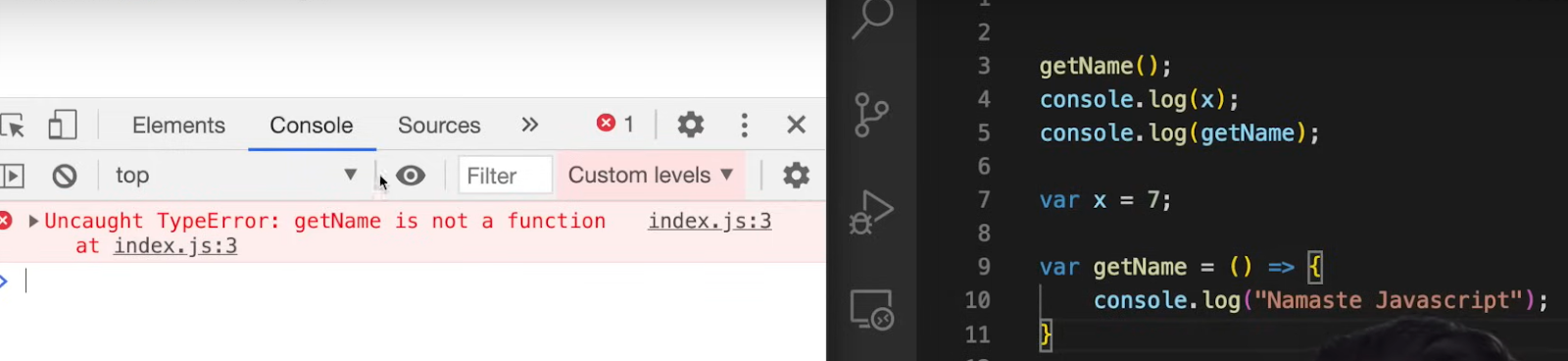
**Even before the first line of execution, in memory creation phase x is undefined and for function whole function block code is copied in memory**

****

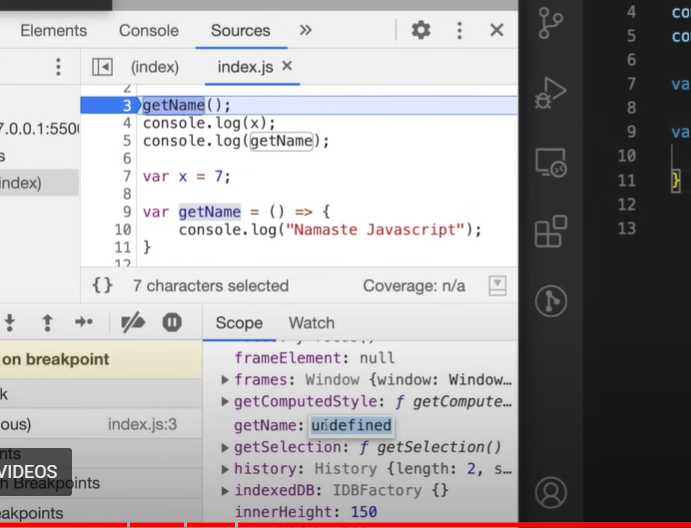


#### Arrow Function

Error on line 3 itself



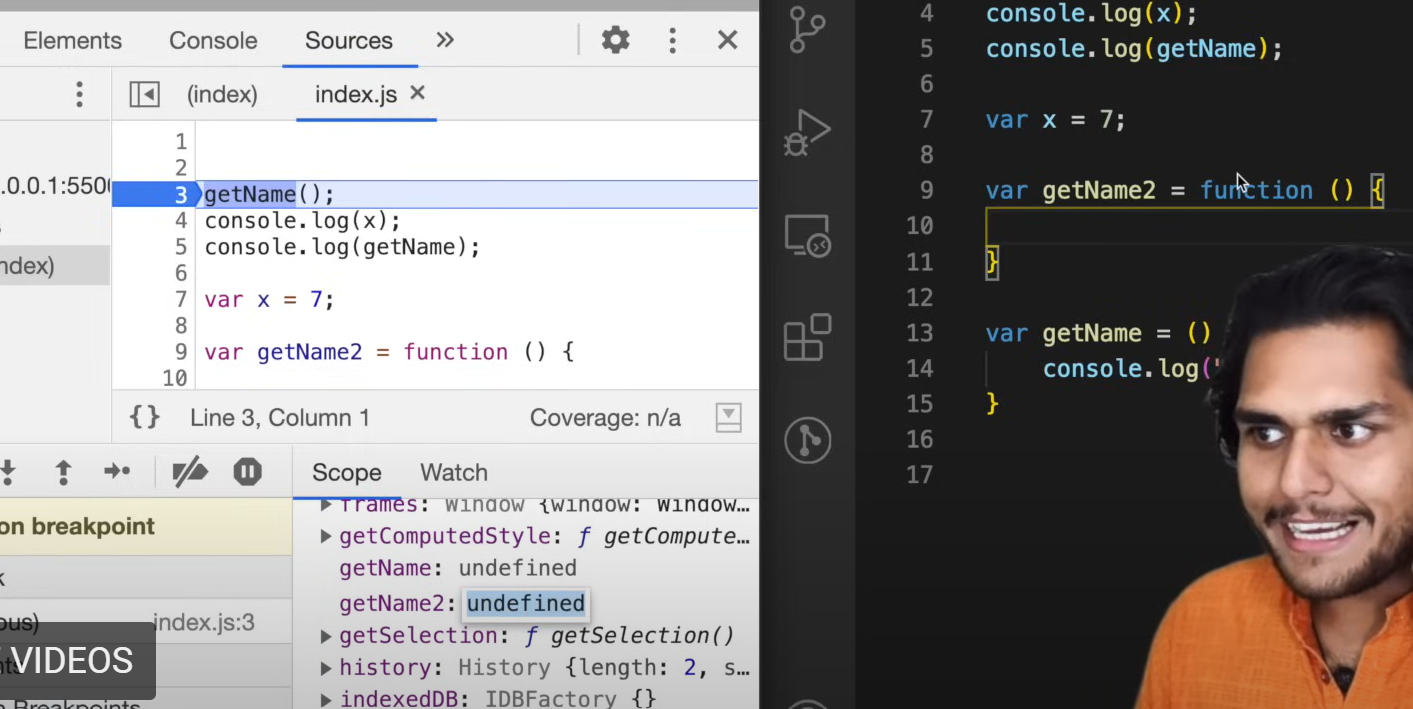
**For arrow function, getName is treated as a variable not a function. Like for variable x, for variable getName also special value ‘undefined’ will be assigned**

****

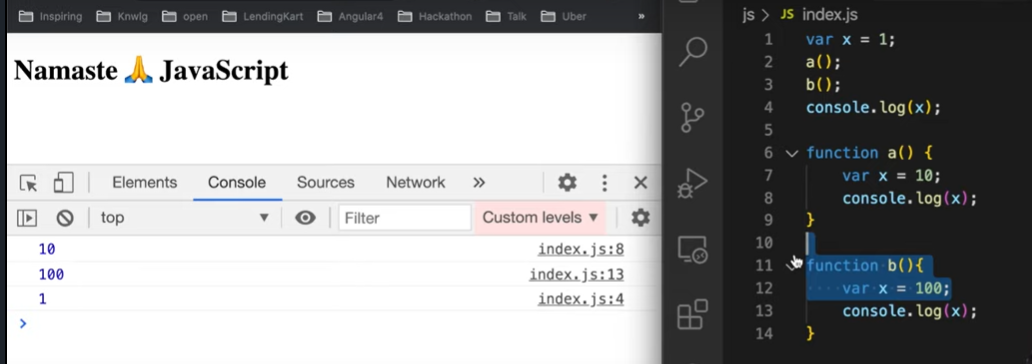
**var getName2 = function () { …. } getName2 is a variable with value undefined**

**var getName = () => { … } getName is a variable with value undefined**

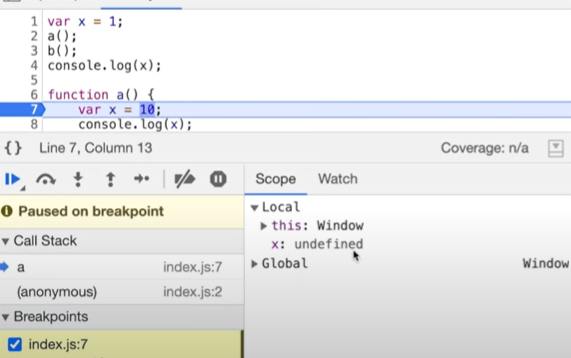
**function getName3() { …. } whole function code will get copied**

****

### Functions in Javascript



#### Local Memory,Global Memory, Global Execution context and a() inside call stack



### Undefined vs Not defined in Javascript

Undefined - special place holder value for variables before it is assigned

Not defined - variable not present in memory

Example

console.log(a); //**undefined**

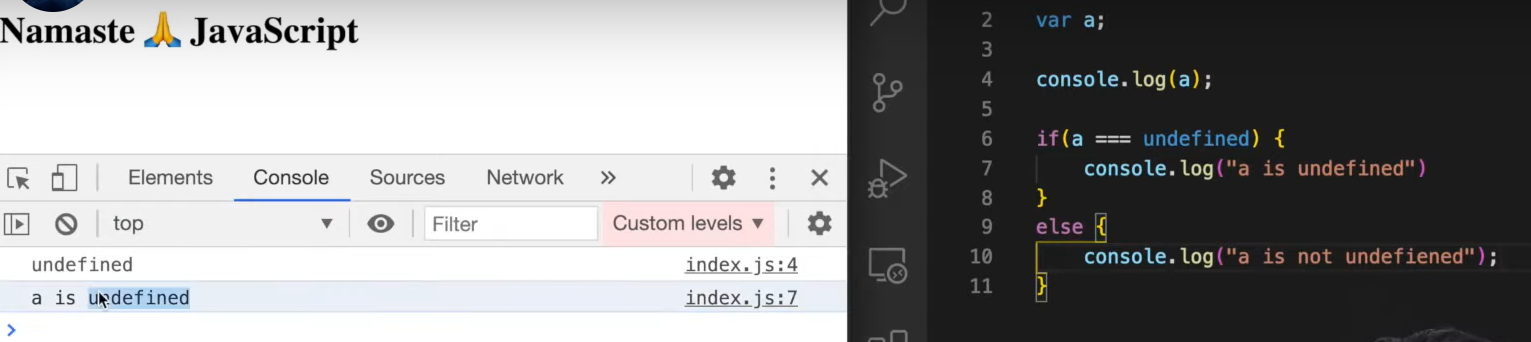
var a = 7;

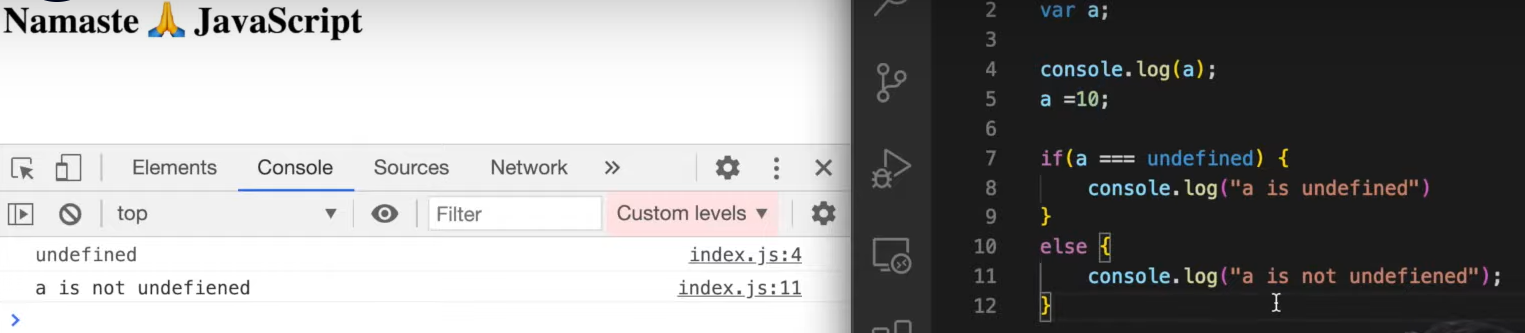
console.log(a); // 7

console.log(x); // Reference Error - x is not defined

var b;

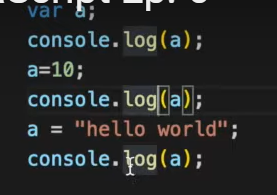
console.log(b); // undefined

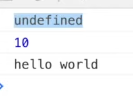




#### Loosely/ weakly typed language

It does not attaches variables to its datatype. U can put string in x and also u can put number in x in later point of time





| var a;  a = undefined; // not error but not a good practice as undefined is a special value for variable  console.log(a) // output is undefined |
| --- |