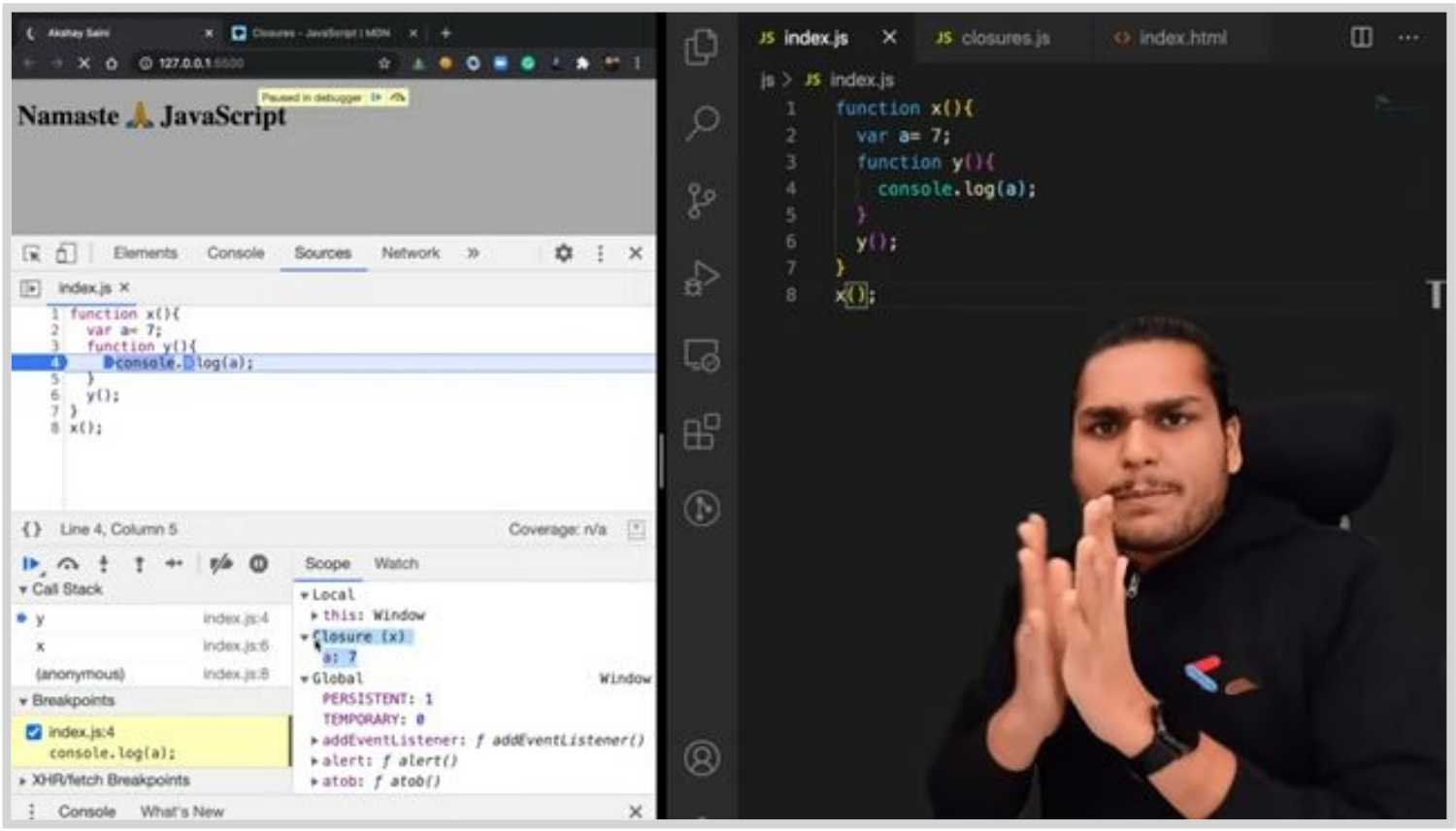
ASKIFY
Make Notes from Youtube Videos

02:21



A function bined together with its lexical environment is called **CLOSURE.**

03:36

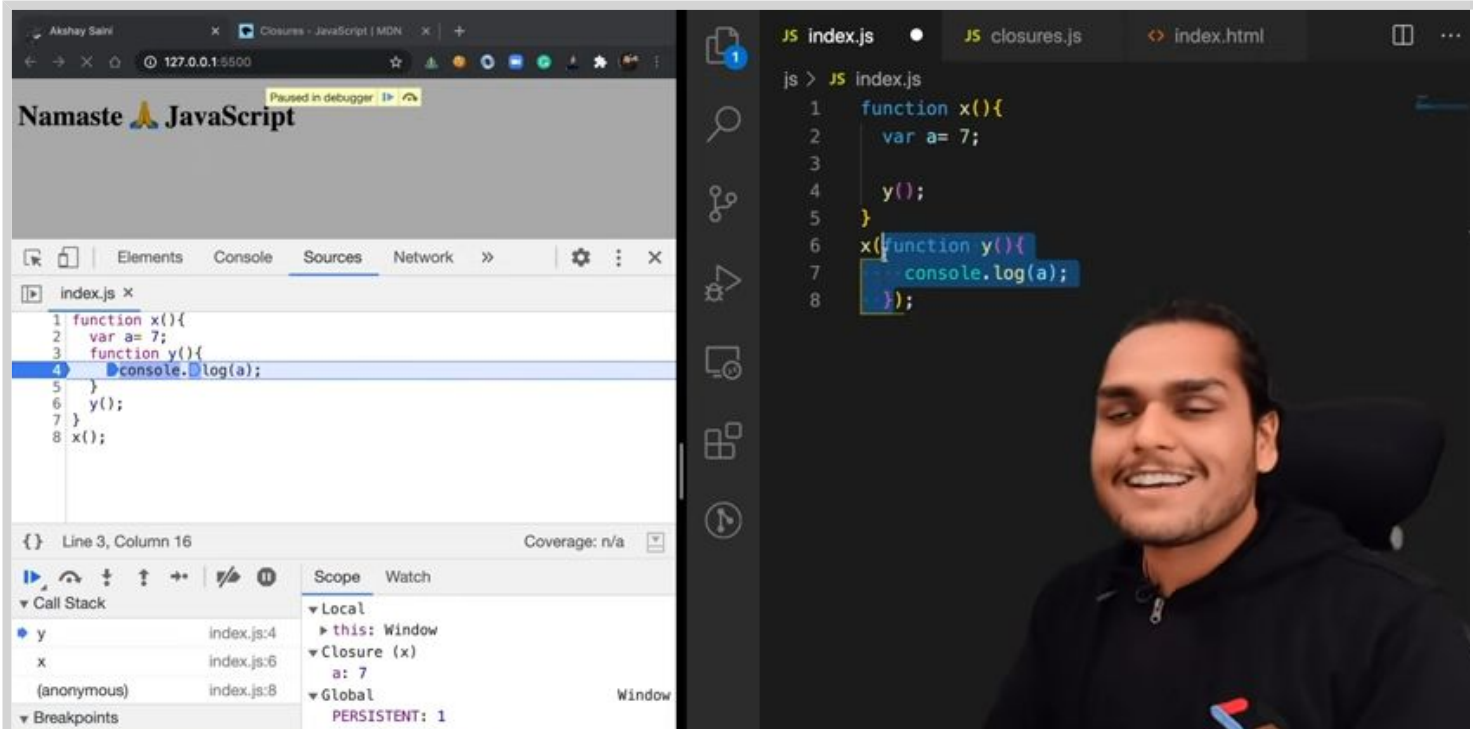
references to its surrounding state (the **lexical environment**). In other words, a closure gives you access to an outer function's scope from an inner function. In JavaScript, closures are created every time a function is created, at function creation time.

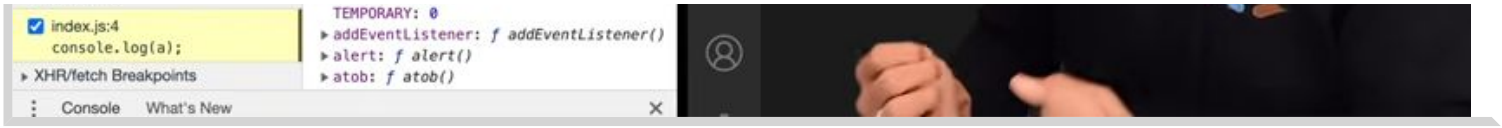
Lexical scoping

Consider the following example code:

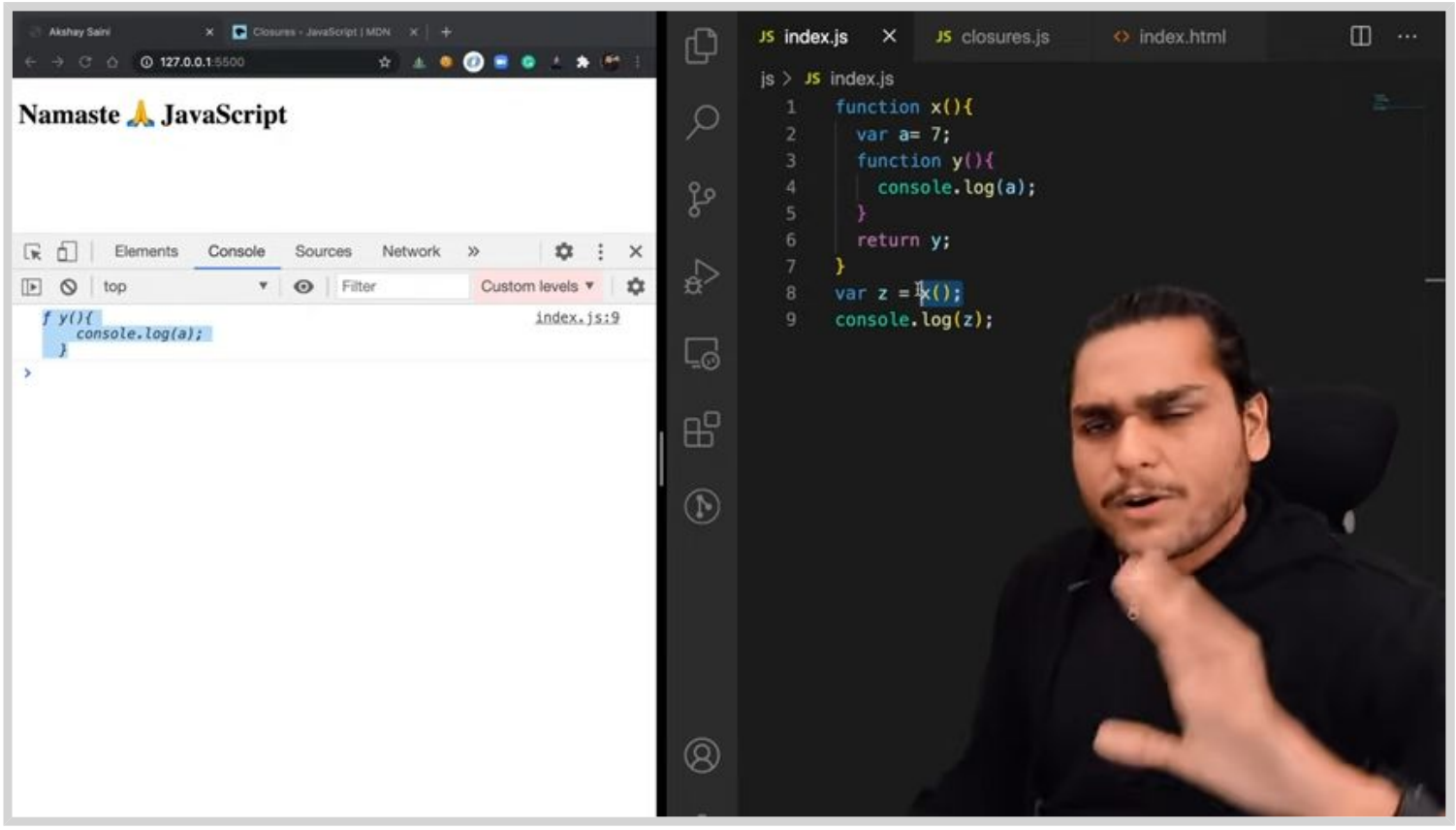
```
1  function init() {
2    var name = 'Mozilla'; // name is a local variable
3    function displayName() { // displayName() is the i
4      alert(name); // use variable declared in the par
5    }
6    displayName();
```

04:32



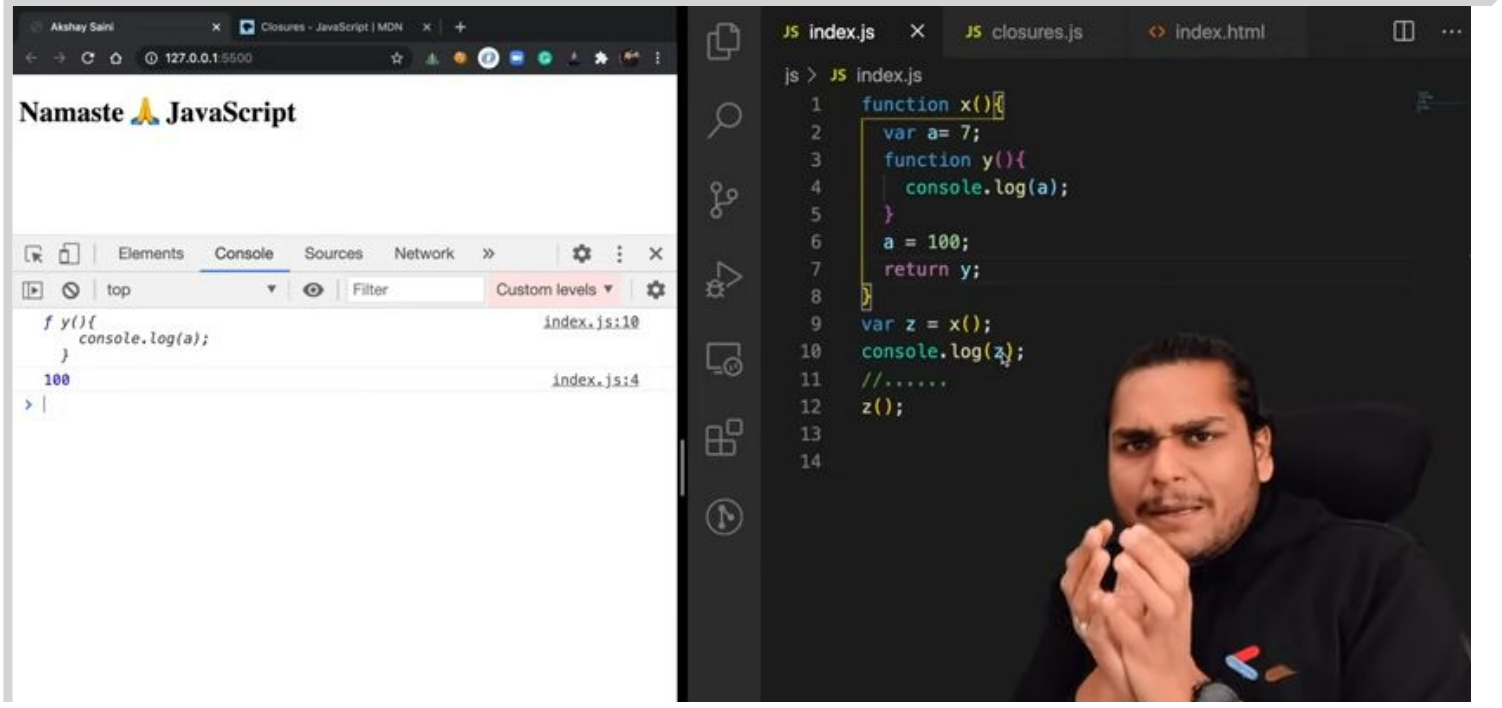


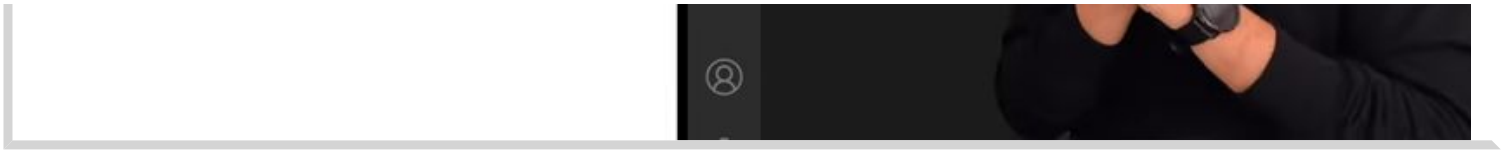
06:26



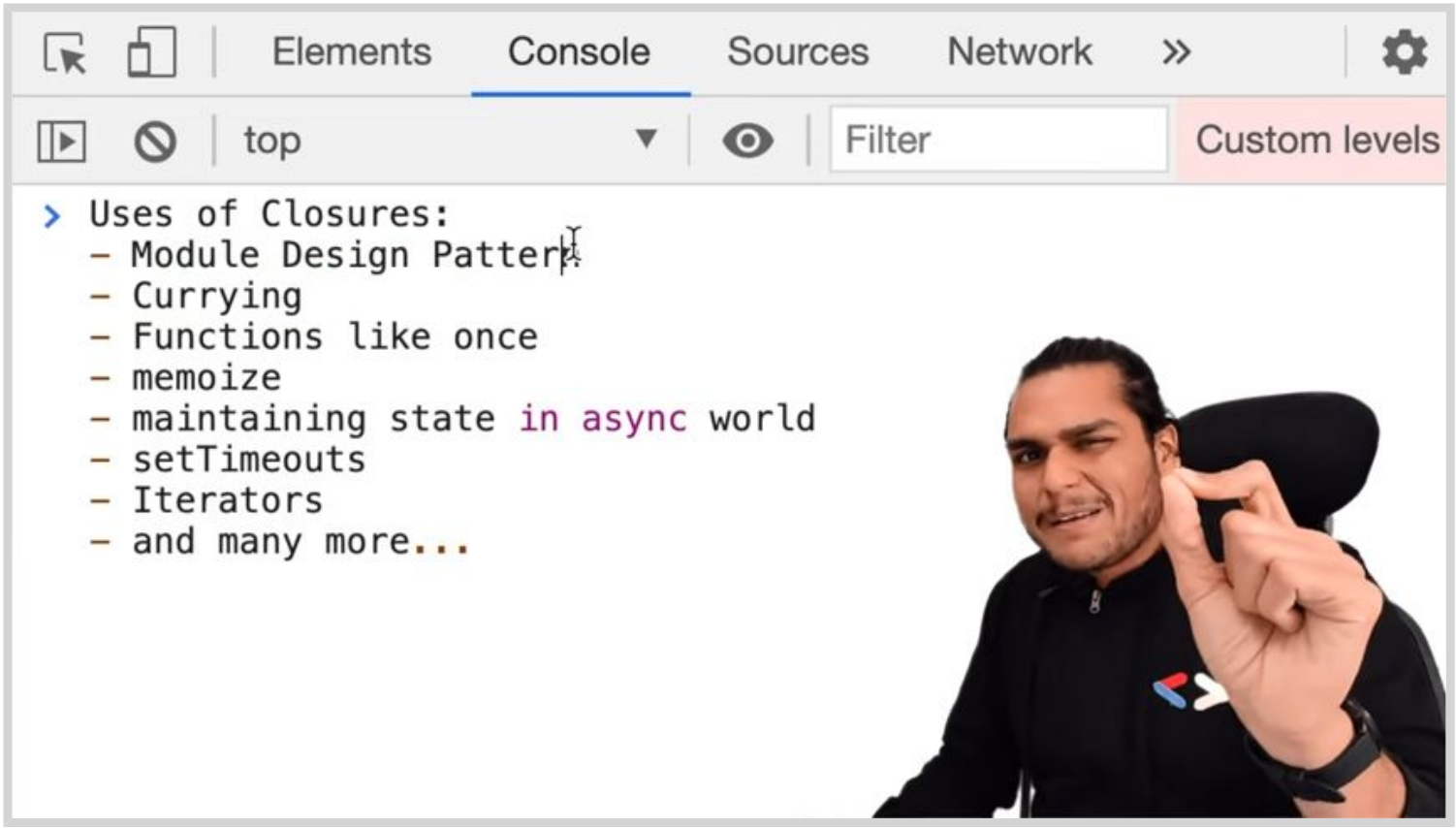
funtion along with its lexical scope returns

15:11

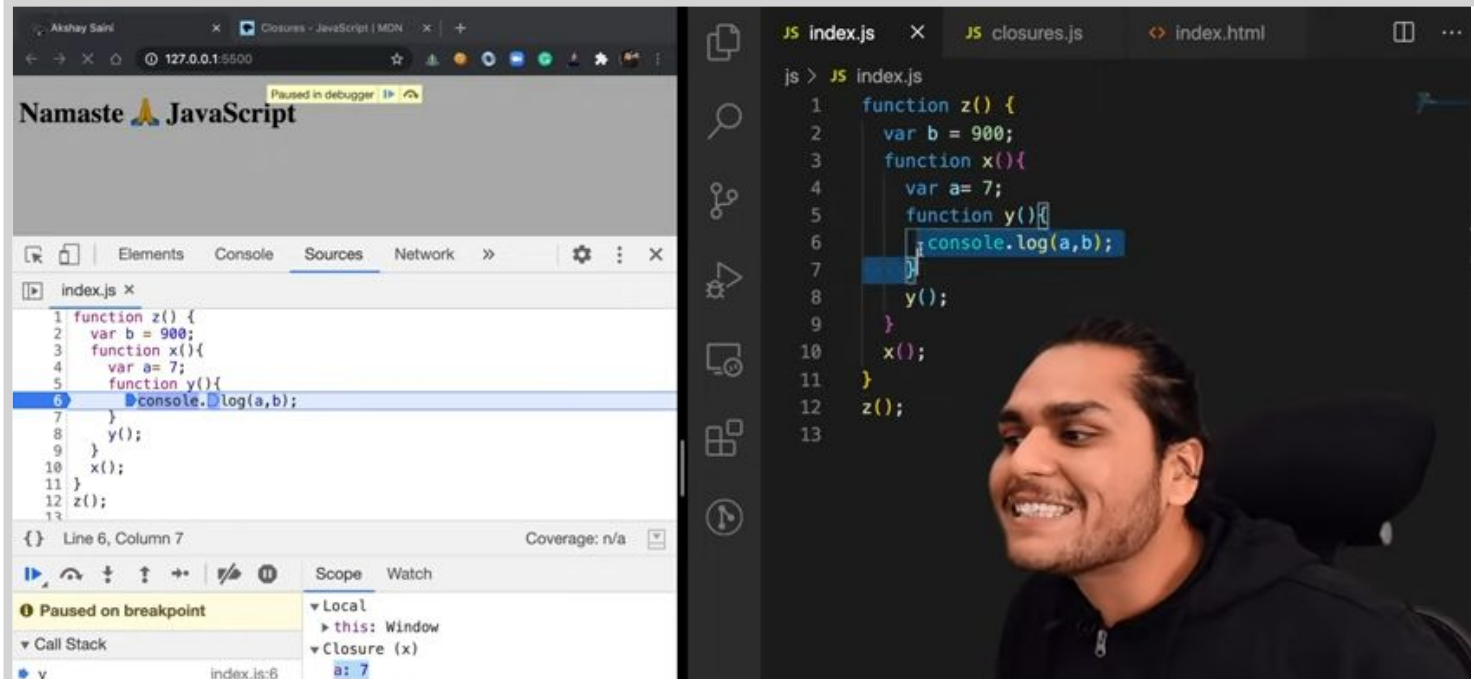


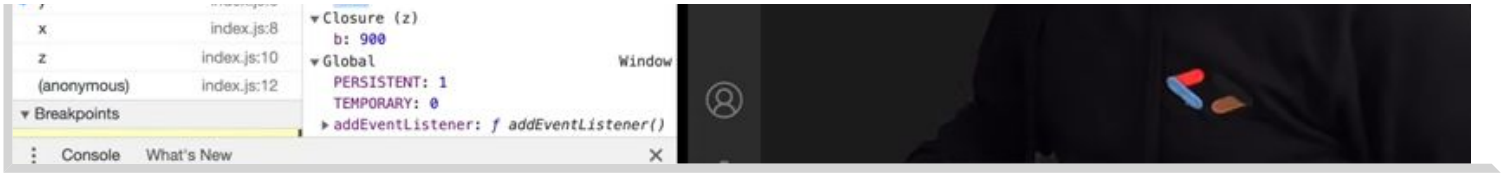


18:52


Elements
Console
Sources
Network
top
Filter
Custom levels
> Uses of Closures:
- Module Design Pattern
- Currying
- Functions like once
- memoize
- maintaining state in async world
- setTimeouts
- Iterators
- and many more...

18:05


Namaste JavaScript
Elements
Console
Sources
Network
index.js
function z() {
var b = 900;
function x(){
var a= 7;
function y(){
console.log(a,b);
}
y();
}
x();
}
z();
Line 6, Column 7
Coverage: n/a
Scope
Watch
Paused on breakpoint
Call Stack
Local
this: Window
Closure (x)
a: 7
index.js
closures.js
index.html


x index.js:8
z index.js:10
(anonymous) index.js:12
Breakpoints
Console
What's New
Closure (z)
b: 900
Global
Window
PERSISTENT: 1
TEMPORARY: 0
addEventListener: ƒ addEventListener()