→ Assigned to variable

e.g. var a = function (param) {
Returned from another function ← return function xyz() {
console.log ('FCF')
}
}

a(function () {
:
}
→ Passed as an argument

→ When we use let, const instead of var, it behaves the same way.

Episode - 14

✱ Callback Functions ft. Event Listeners

→ A callback is a function passed as an argument to another function

→ Functions are first class citizens i.e. take a function A and pass it to another function B. Here, A is callback function. So, basically we are giving access to function B to call A.

→ Callback functions give access to whole Asynchronous world in Synchronous world.

→ JS is synchronous and single threaded language. But we can do async things due to callbacks.

e.g.
```
setTimeout(function() {
    console.log("Timer")
}, 1000)
```
callback → Timer (1s)

→ If any operation blocks the call stack, it is called blocking the main thread

→ Event Listenser : A procedure in JS that waits for an event to occur.

e.g. click events on button

e.g. Counter button

```
<button id="clickMe"> Increment </button>
```

index.js

```
// use closure for data abstraction

function eventListener() {
    let count = 0
    document.getEleById("clickMe")
    .addEventListener("click", function xyz() {
        console.log("Button clicked", ++count)
    })
}
```

Pg No :
Date: / /20

eventListener()

→ Garbage Collection & remove Event Listeners

- Event Listeners are heavy as they form closures.
- So, even when the call stack is empty, Event listeners won't free up memory allocated to count, because it doesn't know when someone will click again.
- So, we remove Event Listeners when we don't need them. (garbage collected)
- onClick, onScroll, onHover all in the page can slow it down heavily.

→ If a function takes more time to run, JS has to wait for it to finish as it only has 1 call stack/ main thread. This blocks the main thread

→ So, always use async functions e.g. setTimeOut when the function has to take more time to run.

→ Never block the main thread.

→ Everything executed in the page is executed through call stack only.