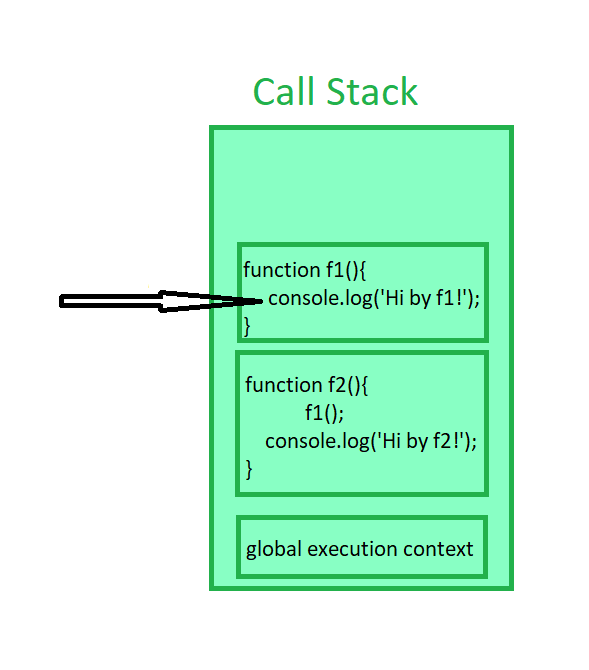
**What is call stack in js?**

* It is a stack in data structure, which works on LIFO (last in fast out) principle.
* When we run our code a GEC ( global execution context ) is created and pushed to call stack.
* GEC initialize memory to variables and functions.
* On every function call a new EC ( execution context ) created and pushed to the call stack.
* As soon as execution over for that function the EC gets deleted.



**What is hoisting?**

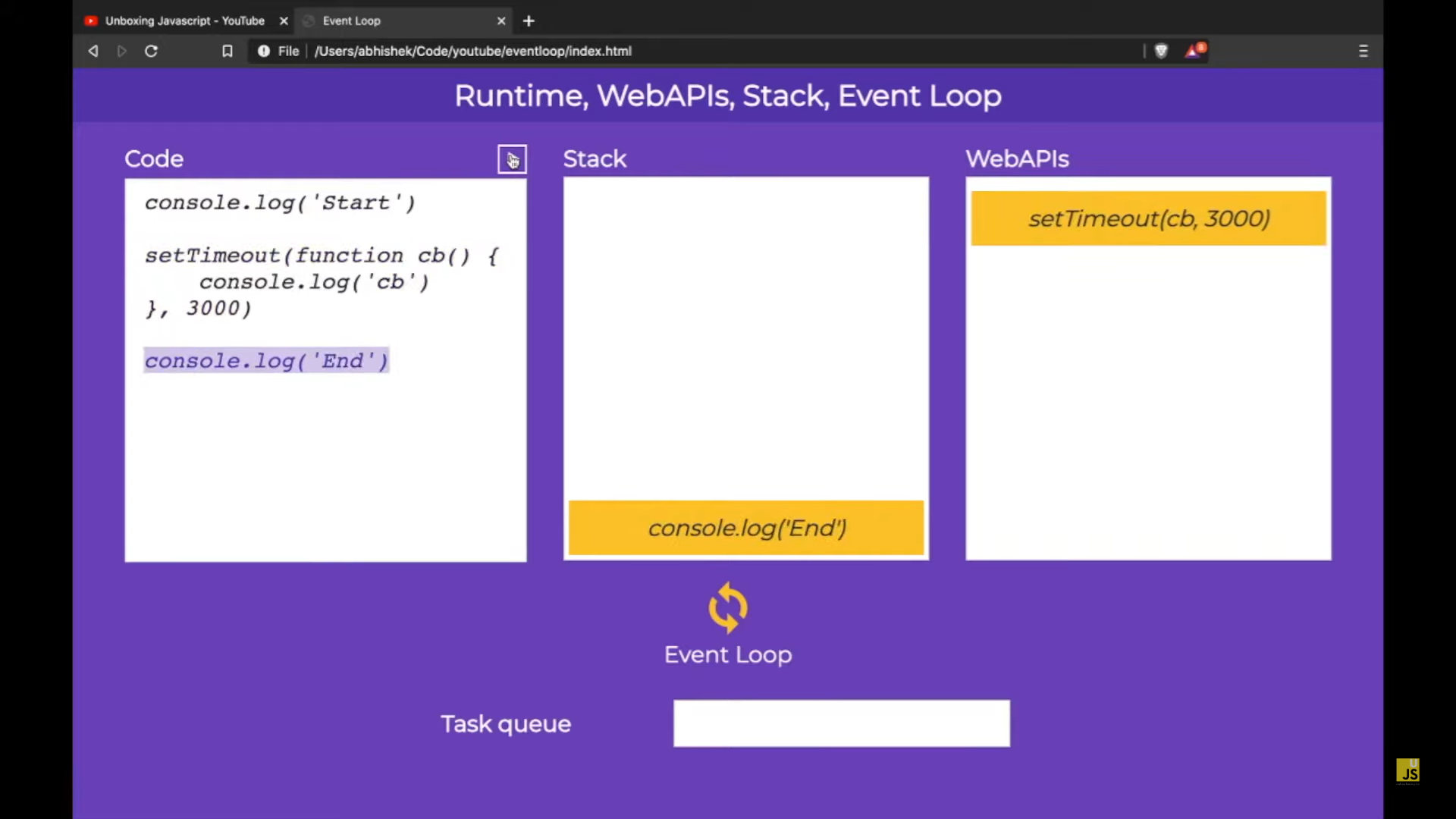
* When we run our code
* GEC ( global execution context ) created and pushed to call stack.
* GEC initialize memory to variables and arrow functions as “**undefined**”.
* GEC initialize functions with function body.

That concept we called as hoisting.

**Var vs let vs const ?**

* “var” is function scoped, means can’t be accessed outside function.
* After var get initialized to “undefined” in global execution context, it can be accessed before initialized to any value.
* Let and const are block scoped, means can’t be accessed out side the block.
* After let and const are initialized to “undefined”, it resides in TDZ (temporal dead zone) and the value can’t be accessed until it initialized to a value.

**Event loop:**



* Call stack can execute one task at a time.
* If that task has a nested function then the inner most function will execute 1st, as call stack works on LIFO principle.
* Asynchronous code or time taking code will propagated to webAPI instead of execting in stack, once result is achived, they will be forwarded to “task Queue”.
* Then event loop will send the result to call stack once it is empty.
* Event loop is something that put result of asynchronous task from task queue to call stack.