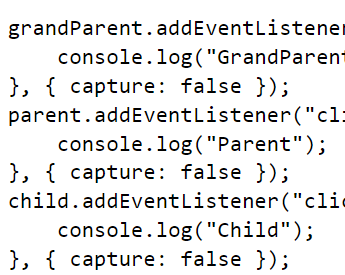
Javascript interview question

Event bubbling and event capturing –

* Event capturing means propagation of event is done from parent elements to child element in the DOM while event bubbling means propagation is done from child element to parent elements in the DOM.
* The event capturing occurs followed by event bubbling.
* If {capture: true} ,event capturing will occur else event bubbling will occur.
* Both can be prevented by using the **stopPropagation()** method.

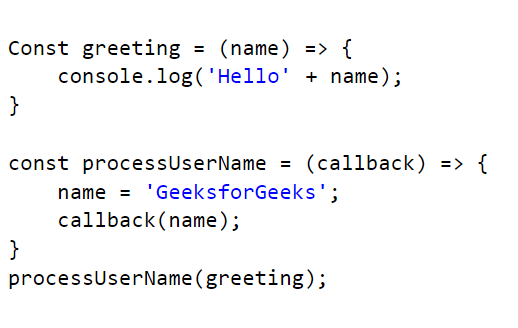


What is scope –

* **Local Scope** allows access to everything within the boundaries (inside the box)
* **Global Scope** is everything outside the boundaries (outside the box). A global scope can not access a variable defined in the local scope because it is enclosed from the outer world, except if you return it.
* **Block Scope**is everything inside the boundaries but it works only for let and const keywords. It does not work with the var keyword.

Immediately invoked Function element

* As the name suggests[IIFE](https://www.geeksforgeeks.org/immediately-invoked-function-expressions-iife-in-javascript/)is a function in Javascript which immediately invoked and executed as soon as it is defined. Variables declared within the IIFE cannot be accessed by the outside world and this way you can avoid the global scope from getting polluted. So the primary reason to use IIFE is to immediately execute the code and obtain data privacy.
* Just like callback function.
* A computer screen shot of a code

  Description automatically generated
* Callback function –
* In javascript, a [callback](https://www.geeksforgeeks.org/javascript-callbacks/) is simply a function that is passed to another function as a parameter and is invoked or executed inside the other function. Here a function needs to wait for another function to execute or return a value and this makes the chain of the functionalities (when X is completed, then Y is executed, and it goes on.). This is the reason callbackis generally used in the asynchronous operation of javascript to provide the synchronous capability.
* 

Closures –

To maintain privacy

Consider a counter function

If we use global variables to access in different function , then it can be change by any function

If we use local function it will not update as required

So we use function inside function as return .

Let var const ==

All three are hoisted but var initialize with undefined value and let and const are initialize with any value ,they are in temporal dead zone.