# Callbacks to Promises

There are many articles on converting javascript callbacks to Promises. For ex: <https://stackoverflow.com/questions/22519784/how-do-i-convert-an-existing-callback-api-to-promises>

So why one more? Ok. Maybe this is better! Or at least one more choice..

Most of the solutions have you coding the callback inside the promise. The method described here takes the callback function and returns both a modified callback, along with a linked promise. This promise resolves when that modified callback is executed. This helps when a callback has to be passed to an external module. Instead of the original callback, we pass the modified callback. The linked promise can then be awaited, and will resolve when the modified callback is called.

In the majority of the cases, it is sufficient to always resolve a promise with a result. The value of the result can be overloaded for error or success. The promise then always resolves successfully and the error condition is indicated by the value of the result. Oftentimes this is much easier than rejecting a promise and handling the reject exception in the await. Assuming this to be the case, the code to implement the conversion of callbacks to promises becomes as simple as the function below:

//======================================

function getLinkedPromiseAndMcb(cb) {

//get the promise to await and the modified cb that will

//fulfill the promise

//Glossary: r=resolve fcn, lp=linked promise, Mcb=modified callback

let r=null //declared here so it non-local to functions in this scope

let lp=new Promise((resolve,reject) => {

r=resolve

})

let Mcb =(...args)=> {

r(cb(...args))

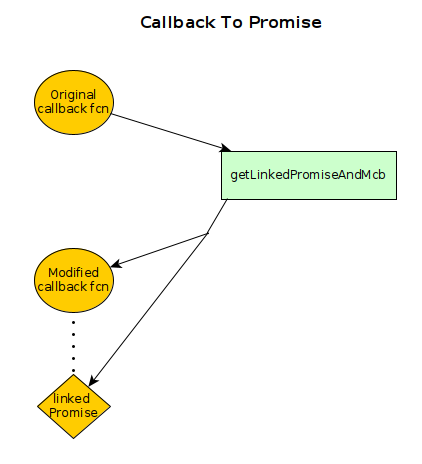
}

return [lp,Mcb]

}

//========================================

This function getLinkedPromiseAndMcb takes a callback function as a parameter and returns a modified callback function (Mcb) along with a promise linked to that Mcb (called lp in the code). A visual representation is:



The Mcb can be used as the callback function to be passed to some external module. The linked pomise (lp) will resolve when the Mcb is called ‘back’ from some external module. The usage of getLinkedPromiseAndMcb is:

//==========================================

//cb is the original callback function

let [lp,mcb] = getPromiseAndMcb(cb)

//start the time-consuming ‘external’ code that eventually calls the mcb

let rslt = await lp; //the lp promise resolves with value returned by the callback (cb)

//==========================================

Here is a complete sample use that can be executed in nodejs (The current nodejs includes the async module):

====================nodejs example================

function getLinkedPromiseAndMcb(cb) {

//get the promise to await and the modified cb that will

//fulfill the promise

//Glossary: r=resolve fcn, lp=linked promise, Mcb=modified callback

let r=null //declared here so it non-local to functions in this scope

let lp=new Promise((resolve,reject) => {

r=resolve

})

let Mcb =(...args)=> {

r(cb(...args))

}

return [lp,Mcb]

}

//sample testing function

async function tester() {

let cb=(rslt)=>{console.log(`Result = "${rslt}" received`)} //sample callback

let [lp,Mcb] = getLinkedPromiseAndMcb(cb)

//simulate an external process returning after 5000 ms

let rt=5000

setTimeout(()=>Mcb('callback Rslt'),rt) //simulates external process calling back cb with a result

console.log("Awaiting callback")

await lp

console.log('done')

}

tester()

//save the above example in a file (‘x.js’)

============ Testing the sample===========

node x.js

//output shown in bold below

**Awaiting callback**

//after a wait of 5 seconds, callback returns

**Result = "callback Rslt" received**

**done**