The Error-first-Callback is quite common style that we use for loadScript.

loadScript('/my/script.js', function(error, script) {

if (error) {

// handle error

} else {

// script loaded successfully

}

});

The convention is:

1. The first argument of the callback is reserved for an error if it occurs. Then callback(err) is called.
2. The second argument (and the next ones if needed) are for the successful result. Then callback(null, result1, result2…) is called.

At first glance, it looks like a viable approach to asynchronous coding. And indeed it is. For one or maybe two nested calls it looks fine.

In the Below code , calls become more nested and the code becomes more dependent and very difficult to manage if there is a loop, conditionals or try…catch etc.

This is termed as “Callback Hell” or “Pyramid of Doom”