Decouple JavaScript Function Call to Multiple Callbacks without Changing Caller or Callee

Ivan Perevernykhata, 22 Oct 2014 CPOL

3.67 (3 votes)

Rate:

A bit of street magic with JavaScript closures and jquery callbacks

Introduction

Let's imagine that you have some legacy or third party code, that has well-known, but awful API. In my case, it was js object that called my functions directly by function name, without support for callbacks.
So if I needed receive some notification from this object, I was required to directly assign appropriate function, e.g.:

Hide Copy Code

```
// 'api' - is that nasty object
api.OnSomeEvent = function (argument){
// my logic
}
```

Sometimes, I needed to receive these callbacks in multiple places, so I was expected to waste my time creating proxy objects that receive notifications from API and transfer them to end destination. It may look like:

Hide Copy Code

```
var proxy = {};
proxy.Callbacks = $.Callbacks();
proxy.OnSomeEventHandler = function(argument){
  proxy.Callbacks.fire(argument);
}

api.OnSomeEvent = proxy.OnSomeEventHandler;

// ...
// ... here is some code that adds callbacks to proxy.Callbacks ...
// ...
```

FYI, this is not real code, I know that my proxy object is miserable, and code is ugly, but this is the simplified version for quick understanding.

What is worse, sometimes API does not provide events that I needed, and I had no chance to change API. So I need to 'hook' required functions (thank goodness they were accessible), e.g.:

Hide Copy Code

```
var proxy = {};
proxy.Callbacks = $.Callbacks();
proxy.OnStupidFunctionHook = function(argument){
  proxy.Callbacks.fire(argument);
  proxy.HookedStupidFunction();
}

proxy.HookedStupidFunction = api.StupidFunction;
api.StupidFunction = proxy.OnStupidFunctionHook;

// ...
// ... here is some code that adds callbacks to proxy.Callbacks ...
// ...
```

Using the Code

Certainly, I was unhappy to write sheets of code for such a trivial task. But JavaScript has an ace in the hole - closures. With such a great instrument, I managed to solve each of the previously described problems with one line of code:

Hide Copy Code

```
JsUtils.SetCallbackHook(api.StupidFunction, function(arguments){ /* I am the callback ! */ });

JsUtils.SetCallbackHook(api.OnSomeCallback, function(arguments){ /* I am the callback ! */ });

JsUtils.SetCallbackHook(api.OnSomeCallback, function(arguments){ /* I am the another one callback ! */ });
```

But let's take a look at what lays under the bonnet of `JsUtils.SetCallbackHook`:

Hide Copy Code

```
var JsUtils = new function() {
    var self = this;

    self.SetCallbackHook = function (sourceFunction, targetCallback) {
        var callBackClosure = function () {
            sourceFunction.callbacks.fireWith(this, $.makeArray(arguments));
        };
        // 1. if function was undefined before - just passing closure function
        if (!sourceFunction) {
            sourceFunction = callBackClosure;
        }
        // 2. if function was defined before, and it is note our closure - hook it, and assign closure
        if(sourceFunction.toString() != callBackClosure.toString()){
            sourceFunction = self.SetCallbackHook(callBackClosure, sourceFunction);
        }
        // 3. make sure our closure has callback
        if (!sourceFunction.callbacks) {
            sourceFunction.callbacks = $.Callbacks();
        }
        // 4. make sure our closure has callback
        sourceFunction.callbacks.add(targetCallback);
        return sourceFunction;
    };
};
```

As you understand from the comments, we replace native function with our closure, that stores also list of callbacks. And each time we assign new callback it is added to the list, and raised once closure is called.

I did not wrap JsUtils to namespace to avoid additional confusion that beginners may feel. You may do it in your own project if need be.

You may play with the working example in this fiddle.

License

Share

About the Author

Ivan

Perevernykhata

Software Developer Unwind Technology
Ukraine

No Biography provided

Comments and Discussions

My vote of 2 — majid torfi — 23-Oct-14 9:11

it is good idea

Reply · Email · View Thread · Permalink · Bookmark

Re: My vote of 2 — Ivan Perevernykhata — 23-Oct-14 9:15

Good idea, but vote of 2

Reply · Email · View Thread · Permalink · Bookmark