# Cascading Asynchronous Function Execution (CAFÉ)

**John Bonfardeci**, 9 Aug 2014 CPOL

★★★★★ 5.00 (2 votes)

Rate:

Cascading Asynchronous Function Execution (CAFÉ) is a design pattern for easily managing the execution sequence of asynchronous requests and callbacks within a medium to large JavaScript framework.

**Download demo project - 2 Kb**

## Introduction

Cascading Asynchronous Function Execution (CAFÉ) is a design pattern for easily managing the execution sequence of asynchronous requests and callbacks within a large JavaScript framework.

## Background

On a recent JavaScript framework I was building for custom web approval forms based on MVVM, the form initialization required many asynchronous requests to various web services within the company to gather data including the user's profile, permissions, group memberships, and workflow history. The requests needed to happen in a specific order since each function relied on data from a previously executed function. All of the requests needed to execute asynchronously for optimal user experience.

After several revisions to the framework, I needed to rearrange the execution order of all of these asynchronous requests and add new ones. This quickly became a problem when tracking down the execution sequence and the callback function calls within each.

This problem compelled me to come up with a simple approach to easily rearrange the execution sequence and management of all asynchronous requests, and callbacks – all in one controlling function near the beginning of the framework code. I call this approach Cascading Asynchronous Function Execution or simply, CAFÉ.

## Using the code

Create an instance of Cafe to execute your asynchronous requests as shown below.

Hide Shrink Copy Code

```javascript
//
/*
*   Cascading Asynchronous Function Execution (CAFE)
*   @params: args = {
*       fns: ['fn1, fn2, fn3, ...'],
*       statusListId: 'string'
*   }
*/
function Cafe(args){
    this.fns = args.fns;
    this.statusList = document.getElementById(args.statusListId);
};

Cafe.prototype = {

    /* handles execution of async functions in this.fns */
    cascade: function(){
        if(arguments[0]){
            this.updateStatus(arguments[0]);
        }

        if(this.fns.length == 0){ return; }

        /* execute the next function passing 'this' and any arguments from the previous function
*/
        this.fns.shift()(this, arguments);
    }

    /* simulate execution of asynchronous request for demonstration purposes */
    , ajaxSim: function(args){
        setTimeout(function(){
            args.success();
        }, 2000);
    }

    /* UI feedback - update the status list after each exectution  */
    , updateStatus: function(str){
        var li = document.createElement("li");
        li.innerHTML = str;
        this.statusList.appendChild(li);
    }
};

/* Run sample functions */
(function(){

    /* Create the CAFE instance. Pass references to async functions in array and ID of a list
element */
    var cafe = new Cafe({
        fns: [fn1, fn2, fn3, fn4, fn5],
        statusListId: "StatusList"
    });

    /* initiate the cascade */
    cafe.cascade('Executing asynchronous cascade...');

    /* example functions - replace these functions with your own */
    function fn1(){

        //AJAX request simulator. Replace with your own AJAX code.
        cafe.ajaxSim({
            success: function(){
                cafe.cascade("Fn1 complete."); /* callback to next function */
            }
        });
    };

    function fn2(){
        //Previous function can pass arguments to this function via the arguments property.
        var args = arguments;

        //AJAX request simulator. Replace with your own AJAX code.
        cafe.ajaxSim({
            success: function(){
                cafe.cascade("Fn2 complete."); /* callback to next function */
            }
        });
    };

    function fn3(){

        //AJAX request simulator. Replace with your own AJAX code.
        cafe.ajaxSim({
            success: function(){
                cafe.cascade("Fn3 complete."); /* callback to next function */
            }
        });
    };

    function fn4(){

        //AJAX request simulator. Replace with your own AJAX code.
        cafe.ajaxSim({
            success: function(){
                cafe.cascade("Fn4 complete."); /* callback to next function */
            }
        });
    };

    function fn5(){

        //AJAX request simulator. Replace with your own AJAX code.
        cafe.ajaxSim({
            success: function(){
                /* callback to next function */
                cafe.cascade("Fn5 complete. All asynchronous operations complete.");
            }
        });
    };

})();
```

## Other Ideas for Your Implementation

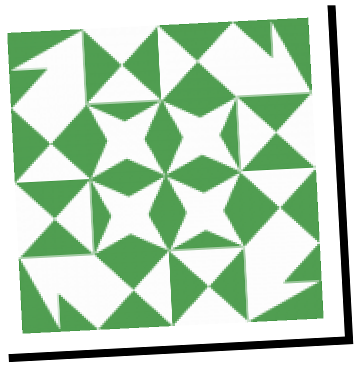
Since the functions are referenced in an array, your approach could be developed further to dynamically change the order of execution or add/remove functions based on any number of circumstances in your implementation using the Array object methods including slice, reverse, sort, push, pop, and shift.

## License

## Share

EMAIL TWITTER

## About the Author



## John Bonfardeci

Software Developer
United States

I'm absolutely addicted to problem solving and I'm fortunate that I can make a living at it by creating original solutions for complex problems in a large health care environment. My areas of focus include design and development on the client, server-side, and database using but not limited to ASP.NET, MVC, C#, SharePoint 2010, SQL, SQL Server 2k-2k8, IIS 6-7.5, JavaScript, jQuery, Knockout JS (MVVM), HTML/5, CSS 2/3, Python, Django, MySQL, SQL Anywhere, and MongoDB. My favorite languages are JavaScript, C#, F#, and Scala - or any language that combines functional and OO paradigms.

## Comments and Discussions

Add a Comment or Question **Search Comments** Go

First Prev Next

**Top job**
**Wombaticus 10-Aug-14 8:01**

Excellent

Reply · Email · View Thread · Permalink · Bookmark

**Re: Top job**
**John Bonfardeci 11-Aug-14 10:17**

Thank you. I'm glad you found it useful.

Reply · Email · View Thread · Permalink · Bookmark

Refresh 1

General News Suggestion Question Bug Answer Joke Praise Rant Admin