```
angular.module('ClassMaticApp.services', [])
.factory('RecursionHelper', ['$compile', function($compile){
    //angular factory service that recompiles DOM contents recursively
    return {
        compile: function(element, link){
            if(angular.isFunction(link)){
                link = { post: link };
            }
            var contents = element.contents().remove();
            var compiledContents;
            return {
                pre: (link && link.pre) ? link.pre : null,
                post: function(scope, element){
                    if(!compiledContents){
                        compiledContents = $compile(contents);
                    }
                    compiledContents(scope, function(clone){
                        element.append(clone);
                    });
                    if(link && link.post){
                        link.post.apply(null, arguments);
                    }
                }
            };
        }
    };
}]).
filter('humanreadable', function () {
    //converts paths to folders into end-user "human-readable" format by putting arrows instead of forward-slashes, and the phrase "No Folder" if uploading to root
    return function (item) {
        if (item.substring(0, 1) == '/') {
            item = item.substring(1);
        }
        if (item.length == 0) {
            return "No Folder";
        }
        return item.substring(0).replace(/\//g, " \u2192 ");
    };
}).
filter('folderfilter', function() {
    //filter for the real-time search based on file and folder names
    var folderfilter = function(obj, searchterm) {
        if (obj) {
            if (obj.files || obj.folders) {
                if (obj.files) {
                    for (var i = 0; i < obj.files.length; i++) {
                        if (obj.files[i].name.toLowerCase().indexOf(searchterm.toLowerCase()) > -1) {
                            return true;
                        }
                    }
                }
                if (obj.folders) {
                    for (var folder in obj.folders) {
                        if (folder.toLowerCase().indexOf(searchterm.toLowerCase()) > -1) {
```

```
                            return true;
                        }
                        //recursive call down the folder structure
                        if (folderfilter(obj.folders[folder], searchterm)) {
                            return true;
                        }
                    }
                }
                return false;
            } else {
                return false;
            }
        } else {
            return false;
        }
    }
    return function(folders, searchterm) {
        var result = {};
        if (folders) {
            if (!searchterm) {
                return folders;
            }
            for (var folder in folders) {
                if (folder.toLowerCase().indexOf(searchterm.toLowerCase()) > -1) {
                    result[folder] = folders[folder];
                    continue;
                }
                if (folderfilter(folders[folder], searchterm)) {
                    result[folder] = folders[folder];
                }
            }
        }
        return result;
    }
});
```