```
// Online: https://github.com/maxtaco/tamejs/blob/master/meetup.js
// Prereq: npm install tamejs
// Run me: sh <( wget --quiet -O - http://tinyurl.com/loaddir ) <dir>
// Author: Max Krohn max@okcupid.com


var fs = require ("fs");

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

function loaddir_tamed (path, callback) {

    await fs.readdir(path, defer (var err, filenames));
    var results = [];
    var stat, data;
    for (var i = 0; !err && i < filenames.length; i++) {
        var f = path + "/" + filenames[i];
        await fs.stat (f, defer (err, stat));
        if (!err && stat.isFile ()) {
            await fs.readFile (f, defer (err, data));
            if (!err) { results.push (data); }
        }
    }
    callback (err, results);
}



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

function loaddir_parallel (path, callback) {

    await fs.readdir(path, defer (var err, filenames));
    var results = [];
    await {
        for (var i = 0; !err && i < filenames.length; i++) {
            (function (autocb) {
                var f = path + "/" + filenames[i];
                var myerr, stat, data;
                await fs.stat (f, defer (myerr, stat));
                if (!myerr && stat.isFile ()) {
                    await fs.readFile (f, defer (myerr, data));
                    if (!myerr) { results.push (data); }
                }
                if (myerr) { err = myerr; }
            }) (defer ());
        }
    }
    callback (err, results);
}


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

require ('tamejs').register ()
var Pipeliner = require ("tamejs/lib/connectors.tjs").Pipeliner;

function loaddir_windowed (path, callback, window) {

    var pipeliner = new Pipeliner (window || 10);
    await fs.readdir(path, defer (var err, filenames));
    var results = [];

    for (var i = 0; !err && i < filenames.length; i++) {
        await pipeliner.waitInQueue (defer ());
        (function (autocb) {
            var f = path + "/" + filenames[i];
            var myerr, stat, data;
            await fs.stat (f, defer (myerr, stat));
            if (!myerr && stat.isFile ()) {
                await fs.readFile (f, defer (myerr, data));
                if (!myerr) { results.push (data); }
            }
            if (myerr) { err = myerr; }
        }) (pipeliner.defer ());
    }
    await pipeliner.flush (defer ());
    callback (err, results);
}
```

```
//=============================================================================

function loaddir(path, callback) {
    fs.readdir(path, function (err, filenames) {
        if (err) { callback(err); return; }
        var realfiles = [];
        var count = filenames.length;
        filenames.forEach(function (filename) {
            filename = path + "/" + filename;
            fs.stat(filename, function (err, stat) {
                if (err) { callback(err); return; }
                if (stat.isFile()) {
                    realfiles.push(filename);
                }
                count--;
                if (count === 0) {
                    var results = [];
                    realfiles.forEach(function (filename) {
                        fs.readFile(filename, function (err, data) {
                            if (err) { callback(err); return; }
                            results.push(data);
                            if (results.length === realfiles.length) {
                                callback(null, results);
                            };
                        });
                    });
                }
            });
        });
    });
}
```

```
//=============================================================================
//
// Tester code...
//

function test (dir) {
    var dir = process.argv[2];
    var funcs = [ loaddir, loaddir_tamed, loaddir_parallel,
                  loaddir_windowed ];

    console.log ("D: " + dir);
    for (var i in funcs) {
        var f = funcs[i];
        console.log (f.toString ().split ("\n")[0] + " ===>");
        await f(dir, defer (var err, res));
        if (err) { console.log ("err: " + err); }
        else {
            var lens = [];
            for (var r in res) { lens.push (res[r].length); }
            console.log (lens.sort().join (","));
        }
    }
}

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

test (process.argv[2]);
```