Current Team Progress with HFOSS Project: mWater Version 3

Reason(s) For Choosing mWater:

We picked the application because we figured that not only would it be a cool thing to test, but the application itself has a pretty universal significance in that it uses image processing to analyze the quality of water, something which each and every one of us needs for survival.

1) We initially started with version 1, realized it hadn't been changed in 3 years. Tried building with android studio (project was dependent on Android SDK). This version included many dependency problems and incomplete files. We moved to version 3 (which we recently discovered).
2) Version 3 written in coffeescript, project cloned, opened with Netbeans. Coffeescript plugin installed. Bryce and I spent some reviewing the language last week, as it's new to both of us.
3) Navigated to Test directory containing 15 different test classes
4) Tried building main project multiple times; no progress. when running test (.coffee) files, they simply open a browser window with their contents copied into it. They don't seem to output any valuable data from test executions.

Getting Started Instructions Listed on V3 Repo

## Getting started

1. Clone this repository
2. Ensure you have Node.js version >= 0.10 installed
3. Ensure you have grunt-cli, browserify, bower and cordova installed globally.
4. Run `npm install` in root folder
5. Run `bower install` in root folder
6. Run `gulp` (version 4+)
7. Run `node server`
8. Visit http://localhost:8080/

I encountered the following problem at step 3, and ha

```
macbook-pro:~ austinhunt$ npm install -g grunt-cli

/usr/local/lib/node_modules/npm/node_modules/npm-package-arg/npa.js:6
let url
^^^
SyntaxError: Unexpected strict mode reserved word
    at Module._compile (module.js:439:25)
    at Object.Module._extensions..js (module.js:474:10)
    at Module.load (module.js:356:32)
    at Function.Module._load (module.js:312:12)
    at Module.require (module.js:364:17)
    at require (module.js:380:17)
    at Object.<anonymous> (/usr/local/lib/node_modules/npm/node_modules/npm-registry-client/lib/access.js:5:11
    at Module._compile (module.js:456:26)
    at Object.Module._extensions..js (module.js:474:10)
    at Module.load (module.js:356:32)
```

I read that this error generally happens when you use a strict mode reserved word as a variable name. The problem, as shown, is on line 6 of the npa.js file, seen here:

```
'use strict'
module.exports = npa
module.exports.resolve = resolve
module.exports.Result = Result

let url
let HostedGit
let semver
let path
let validatePackageName
let osenv

const isWindows = process.platform === 'win32' || global.FAKE_WINDOWS
const hasSlashes = isWindows ? /\\|[/]/ : /[/]/
const isURL = /^(?:git[+])?[a-z]+:/i
const isFilename = /[.](?:tgz|tar.gz|tar)$/i

function npa (arg, where) {
  let name
  let spec
  const nameEndsAt = arg[0] === '@' ? arg.slice(1).indexOf('@') + 1 : arg.indexOf('@')
  const namePart = nameEndsAt > 0 ? arg.slice(0, nameEndsAt) : arg
  if (isURL.test(arg)) {
    spec = arg
  } else if (namePart[0] !== '@' && (hasSlashes.test(namePart) || isFilename.test(namePart))) {
    spec = arg
  } else if (nameEndsAt > 0) {
    name = namePart
    spec = arg.slice(nameEndsAt + 1)
  } else {
    if (!validatePackageName) validatePackageName = require('validate-npm-package-name')
    const valid = validatePackageName(arg)
    if (valid.validForOldPackages) {
      name = arg
    } else {
      spec = arg
    }
  }
  return resolve(name, spec, where, arg)
}

const isFilespec = isWindows ? /^(?:[.]|~[/]|[/\\]|[a-zA-Z]:)/ : /^(?:[.]|~[/]|[/]|[a-zA-Z]:)/

function resolve (name, spec, where, arg) {
  const res = new Result({
    raw: arg,
    name: name,
    rawSpec: spec,
    fromArgument: arg != null
  })

  if (name) res.setName(name)

  if (spec && (isFilespec.test(spec) || /^file:/i.test(spec))) {
    return fromFile(res, where)
"npa.js" 270L, 8129C
```

I’m not sure how to fix this problem, so I haven’t gone beyond this step yet.

Bryce was able to get a little farther than I was (step 5)

Advice recently given to us: Try installing an older version of CoffeeScript; so, since MwaterV3

Set-backs and plans for improvements:
1) Very limited documentation involved in the selected project that we plan to work on.
2) New language adaptation (Coffeescript) that has restricted our ability to fully understand what we are coding.

● Dedicate selective time each day before class to meet for 2 hours and iteratively work on the code, rather than relying on relaying messages back and forth in order to test and run potential ideas.
● Assess the usability of our chosen project and perhaps evaluating acquisition of a new one in order to actually have something tangible.

Tests include:
Test for authorization of user role via - authTest.coffee
Hooks into a database to pass a hardcoded username and password and checks whether it was added to the database successfully or not - RemotedbTests.coffee
Takes the user input picture and tests whether it can make a thumbnail of this image whether a page is opened with this thumbnail presented and whether the image can be removed from the page or not TestQuestions.coffee