GRUNT

Mark Volkmann

mark@ociweb.com

Overview

- JavaScript-based tool for executing many web development tasks
- Runs on Node.js
- Tasks are added via plugins
  - in an earlier version of grunt, there were some built-in tasks, but no more
- There are MANY plugins! ... 957 as of 6/20/13

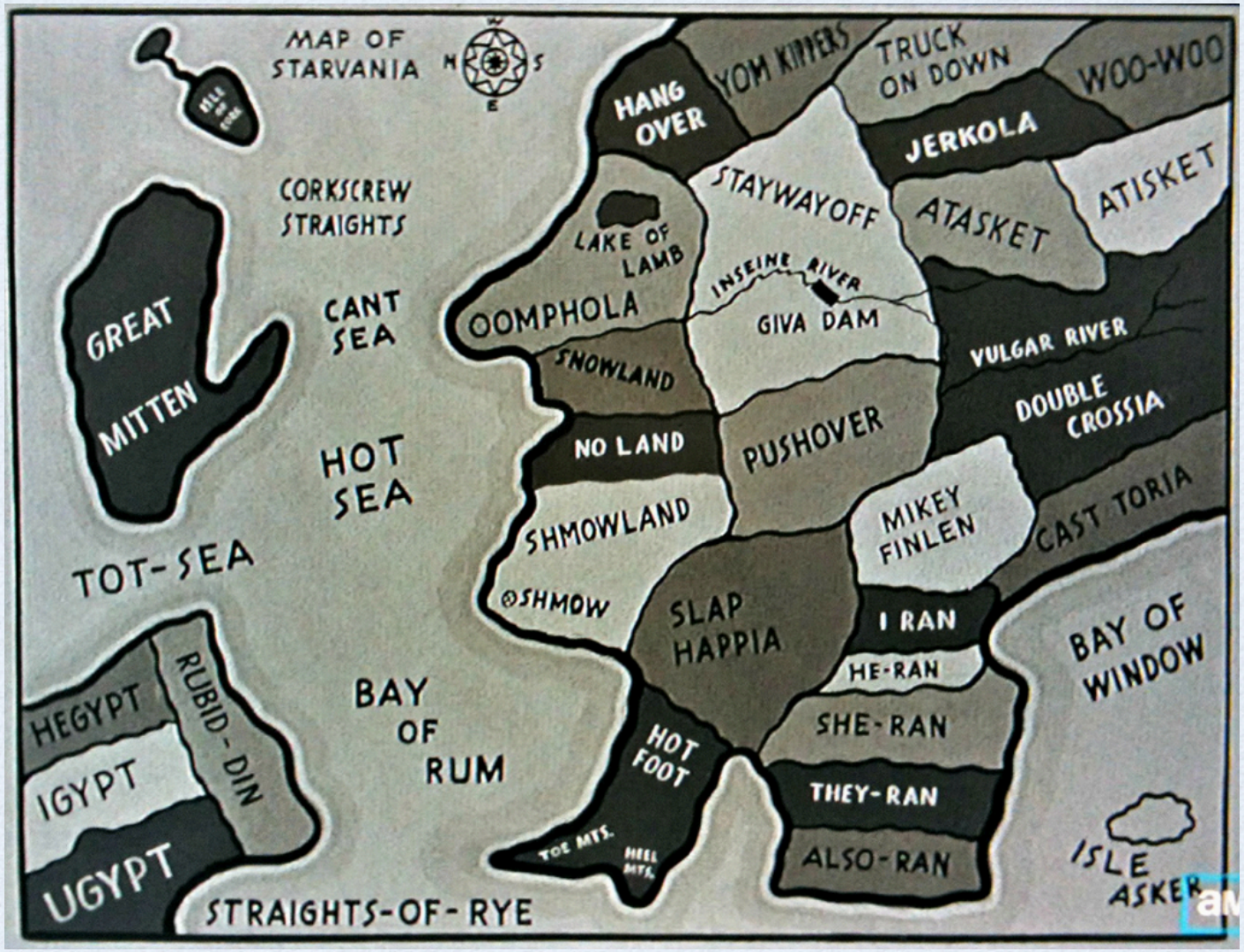
```
npm search gruntplugin | wc -l
```

- http://gruntjs.com

Installing

- **`npm install -g grunt-cli`**
- cd to project directory
- if no **`package.json`** exists yet, create with **`npm init`**
- **`npm install grunt --save-dev`**
  - the **`--save-dev`** option is explained later
  - installs in **`node_modules`** subdirectory
    - can also install globally with **`-g`** so multiple projects can share it
- Create **`Gruntfile.js`** or **`Gruntfile.coffee`**
  - see example later

Three Stooges Maps



Task Examples

- **Beautify** JavaScript
- **Validate**/lint .html, .css, .js and .json files
- **Compile** many kinds of HTML templates
- Compile CoffeeScript to JavaScript
- Compile Compass/LESS/SASS/Stylus to CSS
- **Run** many kinds of **tests** (Jasmine, Mocha, Nodeunit, QUnit, Selenium, Vows, ...)
  - can run browser tests using PhantomJS
- **Generate documentation** from code
- **Concatenate** and **minimize** .js files
- **Deploy** web apps
- **Copy** files to a server using ftp, scp or sftp
- Run **Git** and Subversion commands
- Perform JSON **schema validation**
- Move files to and from **Amazon S3**
- Execute **shell commands**
- **Watch** for file changes and run tasks
  - including reloading affected browser page
- Perform **spell checking**
- Release new versions of NPM projects
- Display **Growl** notifications
- Play **sounds**
- Compile ES6 JavaScript to ES3 JavaScript (uses Google Traceur)
- Compile TypeScript to JavaScript (uses Microsoft TypeScript compiler)
- **Zip** and unzip files

Grunt Plugins Page

Regular vs. Multi-tasks

- **Regular tasks** have one set of configuration options
  - I haven't used any
  - most tasks are multi-tasks
- **Multi-tasks** have one or more named option sets called "targets"
  - for example, the `jshint` task
    takes named arrays of file paths to be checked (can contain wildcards; each is a target)
    and an `options` property that can specify the location of a `.jshintrc` file

Running Grunt

- cd to project directory where Grunt has already been installed
- **`grunt --help`**
  - lists available options and tasks with a description of each
  - "multi-tasks" have an asterisk after description and have subtargets
    - if no subtarget is specified, all are run
- **`grunt options task1 task2 ...`**
  - runs tasks in order specified
  - if no tasks are specified, the default tasks are run
    - if no default is defined, a warning message is displayed and grunt aborts
  - multitasks run all targets unless one is specified with ***`task-name:target-name`***
  - most tasks run to completion and **`grunt`** terminates
  - some run until killed (ctrl-c), such as **`watch`**
  - there are no commonly used options
    - **`--no-write`** performs a "dry run"; no files are created or modified

for demo on my Mac,
```
cdjs
cd grunt-demo
```

we'll see how to define default tasks later

Installing a Plugin

- See searchable list at http://gruntjs.com/plugins
- **`npm install plugin-name --save-dev`**
  - installs in local **`node_modules`** directory
    - can also install globally with **`-g`** so multiple projects can share the plugin
  - **`--save-dev`** option causes **`package.json`** to be modified so **`devDependencies`** property contains a reference to the plugin
  - can delete **`node_modules`** directory and get all plugins back by running **`npm install`**
- Edit Gruntfile
  - add plugin configuration
    - by adding a property to the object that is passed to **`grunt.initConfig`**
    - see plugin documentation for configuration options and examples
  - load plugin near bottom of file
    - **`grunt.loadNpmTasks(plugin-name);`**
  - register any custom tasks

Recommended Plugins

plugins that are published to npm with the "gruntplugin" keyword are automatically cataloged at http://gruntjs.com/plugins

- **grunt** - every Grunt project needs this
- **grunt-clear** - clears terminal window
- **grunt-contrib-coffee** - compiles CoffeeScript files to JavaScript
- **grunt-contrib-less** - compiles LESS files to CSS
- **grunt-html** - validates HTML files; VERY SLOW!
- **grunt-contrib-connect** - runs a local HTTP server for serving static files
- **grunt-contrib-csslint** - validates CSS files
- **grunt-coffeelint** - validates CoffeeScript files
- **grunt-contrib-jshint** - validates JavaScript files
- **grunt-jsonlint** - validates JSON files, like package.json
- **grunt-contrib-clean** - deletes specified directories and files
- **grunt-mocha-cli** - runs Mocha tests
- **grunt-contrib-watch** - described on next slide

grunt-contrib-watch plugin

- Runs specified tasks when new files are created and existing files are modified
- Configure with an object where properties are file extensions and values are objects with **`files`** and **`tasks`** properties whose values are arrays
  - for example

many properties like this take a string file path or an array of them; file paths can contain wildcard characters

```
watch: {
  js: {
    files: ['Gruntfile.js', 'lib/**/*.js', 'test/**/*.js'],
    tasks: ['jshint']
  },
  ...
}
```

- Can configure to refresh current browser window when files are modified
  - see **`livereload`** option in upcoming example
- Don't have to restart after modifying Gruntfile

Gruntfile Structure

```
module.exports = function (grunt) {
  grunt.initConfig({
    // For each plugin task ...
    plugin-task-name: {
      // Configuration properties for the task go here.
    },
    ...
  });

  // For each plugin ...
  grunt.loadNpmTasks('plugin-name');

  // For each custom task ...
  grunt.registerTask('custom-task-name', [dependency-task-names]);
  // Custom tasks can depend on plugin tasks and other custom tasks.
});
```

Gruntfile Example ...

```
module.exports = function (grunt) {
  grunt.initConfig({
    clean: ['build'],
    coffee: {
      all: {
        expand: true,
        cwd: 'lib',
        src: ['*.coffee'],
        dest: 'build/lib',
        ext: '.js'
      },
      options: {
        sourceMap: true
      }
    },
    coffeelint: {
      files: ['lib/**/*.coffee']
    },
```

```
    connect: {
      server: {
        options: {
          port: 3000, // default is 8000
          base: '.' // the default
        }
      }
    },

    csslint: {
      strict: {
        options: {
          ids: false
        },
        src: ['styles/*.css', 'build/styles/*.css']
      }
    },
```

allows ids to be used in CSS selectors

... Gruntfile Example ...

```
htmllint: {
  all: ['*.html']
},
jshint: {
  all: ['Gruntfile.js', 'build/**/*.js',
    'lib/**/*.js', 'test/**/*.js'],
  options: {
    jshintrc: '/Users/Mark/.jshintrc'
  }
},
jsonlint: {
  all: {
    src: ['*.json']
  }
},
```

very slow and doesn't check indentation

```
less: {
  all: {
    files: [{
      expand: true,
      cwd: 'styles',
      src: ['*.less'],
      dest: 'build/styles',
      ext: '.css'
    }]
  }
},
mochacli: {
  options: {
    recursive: true,
    reporter: 'list',
    require: ['chai'],
    ui: 'tdd'
  },
  all: ['test/**/*.js']
},
```

... Gruntfile Example ...

```
watch: {
  options: { livereload: true },
  coffee: {
    files: ['lib/**/*.coffee', 'test/**/*.coffee'],
    tasks: ['coffee']
  },
  css: {
    files: ['styles/*.css'],
    tasks: ['csslint']
  },
  html: {
    files: ['*.html'],
    tasks: ['htmllint']
  },
  js: {
    files: ['Gruntfile.js', 'lib/**/*.js',
      'test/**/*.js'],
    tasks: ['jshint', 'mochacli']
  },
  json: {
    files: ['*.json'],
    tasks: ['jsonlint']
  },
  less: {
    files: ['styles/*.less'],
    tasks: ['less', 'csslint']
  }
} // end of watch config
}); // end of call to grunt.initConfig
```

Add the following to HTML file being browsed:

```
<script src="http://localhost:35729/livereload.js"></script>
```

… Gruntfile Example

```
  var tasks = [
    'grunt-coffeelint',
    'grunt-contrib-clean',
    'grunt-contrib-coffee',
    'grunt-contrib-connect',
    'grunt-contrib-csslint',
    'grunt-contrib-jshint',
    'grunt-contrib-less',
    'grunt-contrib-watch',
    'grunt-html', // This is VERY SLOW!
    'grunt-jsonlint',
    'grunt-mocha-cli'
  ];
  tasks.forEach(grunt.loadNpmTasks);

  grunt.registerTask('all', ['lint', 'coffee', 'less', 'test']);
  grunt.registerTask('default', ['connect', 'watch']);
  grunt.registerTask('lint',
    ['coffeelint', 'csslint', 'htmllint', 'jshint', 'jsonlint']);
  grunt.registerTask('test', ['mochacli']);
};
```

defines tasks run when none are specified

when 2nd arg is an array, **registerTask** defines an "alias task"

Demo Time ...

- **`cdjs; cd grunt-demo`**
- **`grunt clean`**
  - deletes build directory created by previously run commands
- **`grunt coffee`**
  - compiles all **`.coffee`** files in **`lib`** directory to **`.js`** files in **`build/lib`**
  - generates source map files with same name as **`.coffee`** files, but **`.map`** extension
- **`grunt coffeelint`**
  - checks for issues in **`/lib .coffee`** files
    - try removing `)` in first line of **`lib/fromCS.coffee`**
- **`grunt csslint`**
  - checks for issues in **`.css`** files in **`styles`** directory
    - try changing **`red`** to **`rod`** in 2nd line of **`styles/demo.css`**

... Demo Time ...

- **`grunt htmllint`**
  - checks for issues in **`.html`** files in current directory
    - try changing **`<html>`** to **`<htm>`** in 2nd line of **`index.html`**
  - much slower than other plugins; generates some misleading messages
- **`grunt jslint`**
  - checks for issues in **`.js`** files in **`lib`** directory
    - try changing **`red`** to **`rod`** in 2nd line of **`styles/demo.css`**
- **`grunt jsonlint`**
  - checks for issues in **`.json`** files in current directory
    - try removing first **`"`** in 2nd line of **`package.json`**
- **`grunt less`**
  - compiles all **`.less`** files in **`styles`** directory to **`.css`** files in **`build/styles`**
    - generates **`build/styles/fromLESS.css`**

… Demo Time

- **`grunt test`**
  - runs Mocha tests below **`test`** directory
    - modify **`assert`** near end of **`test/demo.mocha.js`** to fail and run this
- **`grunt`**
  - starts local HTTP server then
    watches for changes to files with many extensions,
    runs the appropriate tasks on them,
    and reloads current browser window
    - browse **`index.html`**
    - run "**`grunt clean watch`**", then modify these files, observe terminal output, and look for changes in reloaded browser window
    - **`lib/fromCS.coffee`** - new .js generated and linted
    - **`styles/demo.css`** - lints | change **`h1`** color |
    - **`index.html`** - lints
    - **`lib/demo.js`** - lints, runs tests | break add function in **`demo.js`** and note failing test |
    - **`styles/fromLESS.less`** - new .css generated and linted

Using CoffeeScript SourceMaps

- Currently only works in Chrome
- Steps to enable
  - select View...Developer...Developer Tools
  - click gear icon in lower-right
  - check "Enable source maps"
  - change "Indentation" to 2 spaces ... the only sane value
- Generate a `.js` and `.map` file for each `.coffee` file
  - put them in the same directory
  - grunt-contrib-coffee tasks does this when `sourceMap` option is `true`
  - `coffee` command generates both `.js` and `.map` files when `-c` and `-m` options are specified
- Add script tag for generated `.js` file to `.html`
- View `.coffee` files from Developer Tools "Sources" tab
  - can set breakpoints and step through CoffeeScript code!

Creating a Grunt Plugin

- Setup
  - `npm install -g grunt-init`
  - `git clone git://github.com/gruntjs/grunt-init-gruntplugin.git \ ~/.grunt-init/gruntplugin`
  - create an empty directory for the plugin and cd to it
  - `grunt-init gruntplugin`
    - see the session on the next slide
  - `npm init`
    - to setup the development environment for this plugin
    - installs four node modules locally

```
myplugin$ grunt-init gruntplugin
Running "init:gruntplugin" (init) task
This task will create one or more files in the current directory, based on the
environment and the answers to a few questions. Note that answering "?" to any
question will show question-specific help and answering "none" to most questions
will leave its value blank.

"gruntplugin" template notes:
For more information about Grunt plugin best practices, please see the docs at
http://gruntjs.com/creating-plugins

Please answer the following:
[?] Project name (grunt-myplugin) grunt-filestats
[?] Description (The best Grunt plugin ever.) Outputs statistics for a given file
[?] Version (0.1.0)
[?] Project git repository (git://github.com/Mark/myplugin.git) git://github.com/mvolkman/grunt-filestats.git
[?] Project homepage (https://github.com/mvolkman/grunt-filestats)
[?] Project issues tracker (https://github.com/mvolkman/grunt-filestats/issues)
[?] Licenses (MIT)
[?] Author name (R. Mark Volkmann)
[?] Author email (r.mark.volkmann@gmail.com)
[?] Author url (none) http://ociweb.com/mark
[?] What versions of grunt does it require? (~0.4.1)
[?] What versions of node does it run on? (>= 0.8.0)
[?] Do you need to make any changes to the above before continuing? (y/N) N

Writing .gitignore...OK
Writing .jshintrc...OK
Writing Gruntfile.js...OK
Writing README.md...OK
Writing tasks/filestats.js...OK
Writing test/expected/custom_options...OK
Writing test/expected/default_options...OK
Writing test/fixtures/123...OK
Writing test/fixtures/testing...OK
Writing test/filestats_test.js...OK
Writing LICENSE-MIT...OK
Writing package.json...OK

Initialized from template "gruntplugin".
You should now install project dependencies with npm install. After that, you
may execute project tasks with grunt. For more information about installing
and configuring Grunt, please see the Getting Started guide:

http://gruntjs.com/getting-started

Done, without errors.
```

Editing Plugin Task

- See tips at http://gruntjs.com/creating-tasks
- Choose task type
  - basic task
    - runs given task function without using configuration properties in Gruntfile
    - can pass colon separated arguments after task name in command-line
  - multi-task
    - runs given task function and can access multiple target configurations in Gruntfile
  - custom task
    - runs given task function and can access configuration properties in Gruntfile that do not follow multi-task structure
- Edit **`tasks/`*`name`*`.js`**
  - follow guidelines for selected task type

Basic Task Example

```
module.exports = function (grunt) {
  grunt.registerTask('filestats', 'outputs statistics for a file', function (filePath) {
    if (!filePath) {
      return grunt.log.writeln(this.name + ' must be given a file path');
    }

    var fs = require('fs');

    var done = this.async();

    fs.stat(filePath, function (err, stats) {
      grunt.log.writeln('stats for ' + filePath + ':');
      Object.keys(stats).sort().forEach(function (key) {
        grunt.log.writeln('  ' + key + ' = ' + stats[key]);
      });
      done();
    });
  });
};
```