

# JavaScript Automation

Stop Doing Mundane Work

@craigphares

[craigphares.github.io/javascript-automation/](https://craigphares.github.io/javascript-automation/)

# **As Web Developers, We're Told to Do Lots of Things**

Work in small, logical chunks of code to keep things manageable and organized

Create tests early and often during development (TDD)

Use a preprocessor (like Sass) for CSS to keep code DRY

Compress and minify all static assets to keep load time down

Optimize images to reduce file size without sacrificing quality

**How do we perform these tasks  
outside of a server environment?**

Grunt is a  
**Task Runner**



**As Web Developers,**

# **We Depend on Lots of Things**

Need a layout framework for a quick build?

**Bootstrap, Foundation**

**jQuery**, enough said

Organizing JS into MV\*?

**Backbone.js, Angular, Ember**

Using audio?

**Howler, SoundJS**

**How do we reference these dependencies, while keeping separation from our own source?**

Bower is a  
**Package Manager**





# Our New Workflow



**Bower**

+



**Grunt**

=



**Awesome**



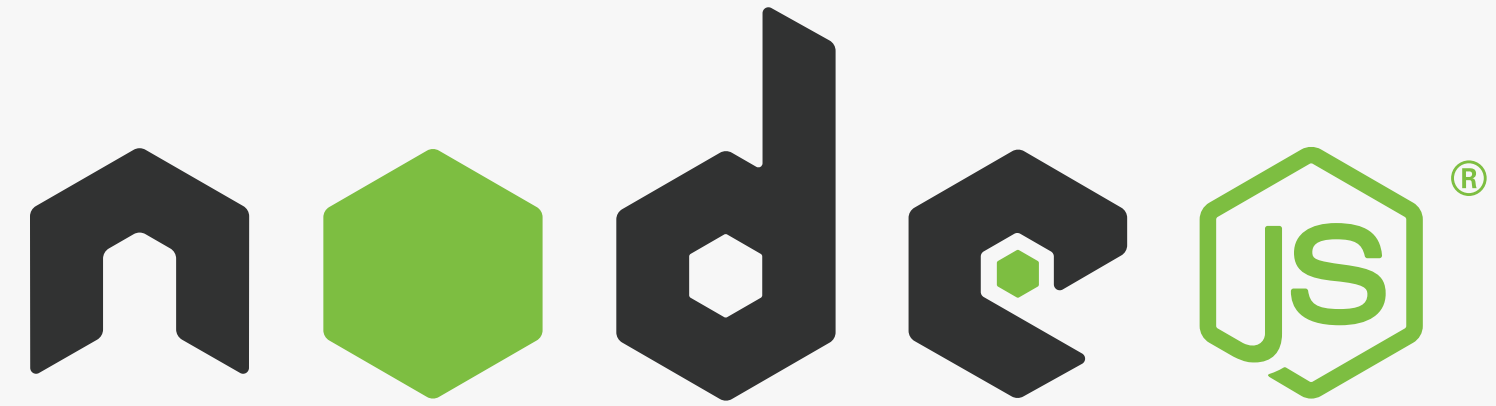
# Stack Setup

This assumes that you're already using Git.  
If not, start using it: [git-scm.com](https://git-scm.com)

You can clone this entire project from GitHub:

```
$ git clone git@github.com:craigphares/javascript-automation.git
```





Bower and Grunt run on Node, but you don't have to know Node to use them.

Install node from their website: [nodejs.org](https://nodejs.org)

Or with Homebrew:  
**\$ brew install node**



Install the Bower command line interface  
**\$ npm install -g bower**



Install the Grunt command line interface  
**\$ npm install -g grunt-cli**



Install Sass  
**\$ gem install sass**

**Onto the Project**

# The package.json File


Node projects require a single package.json file at the root level.

**\$ npm init**

```
{
  "name": "javascript-automation",
  "version": "1.0.0",
  "description": "Stop doing mundane work.",
  "main": "index.html",
  "repository": {
    "type": "git",
    "url": "https://github.com/craigphares/javascript-automation.git"
  },
  "author": "Craig Phares <craigphares@sixoverground.com>",
  "license": "MIT",
  "bugs": {
    "url": "https://github.com/craigphares/javascript-automation/issues"
  },
  "homepage": "https://github.com/craigphares/javascript-automation"
}
```

Name

LICENSE  
package.json  
README.md



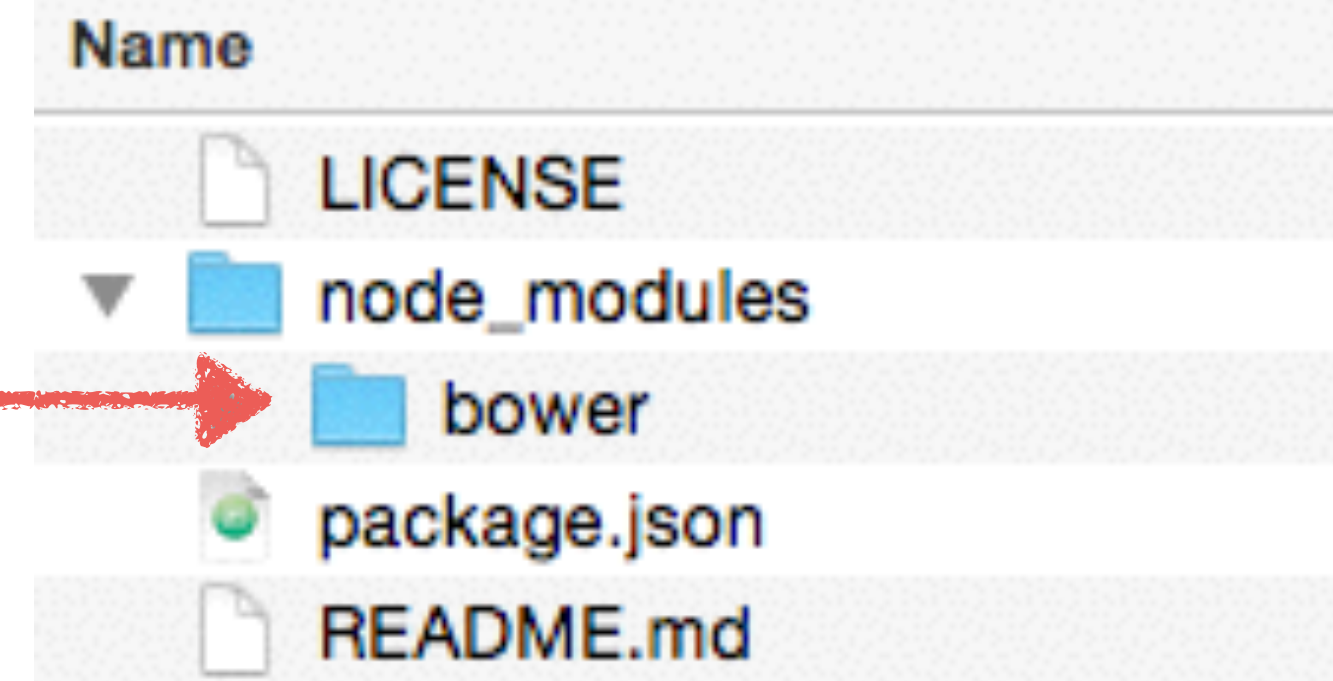


# Let's Install Bower

Node modules are installed in the node\_modules directory.

```
$ npm install bower --save-dev
```

```
{
  "name": "javascript-automation",
  "version": "1.0.0",
  "description": "Stop doing mundane work.",
  "main": "index.html",
  "repository": {
    "type": "git",
    "url": "https://github.com/craigphares/javascript-automation.git"
  },
  "author": "Craig Phares <craigphares@sixoverground.com>",
  "license": "MIT",
  "bugs": {
    "url": "https://github.com/craigphares/javascript-automation/issues"
  },
  "homepage": "https://github.com/craigphares/javascript-automation",
  "devDependencies": {
    "bower": "^1.4.1"
  }
}
```



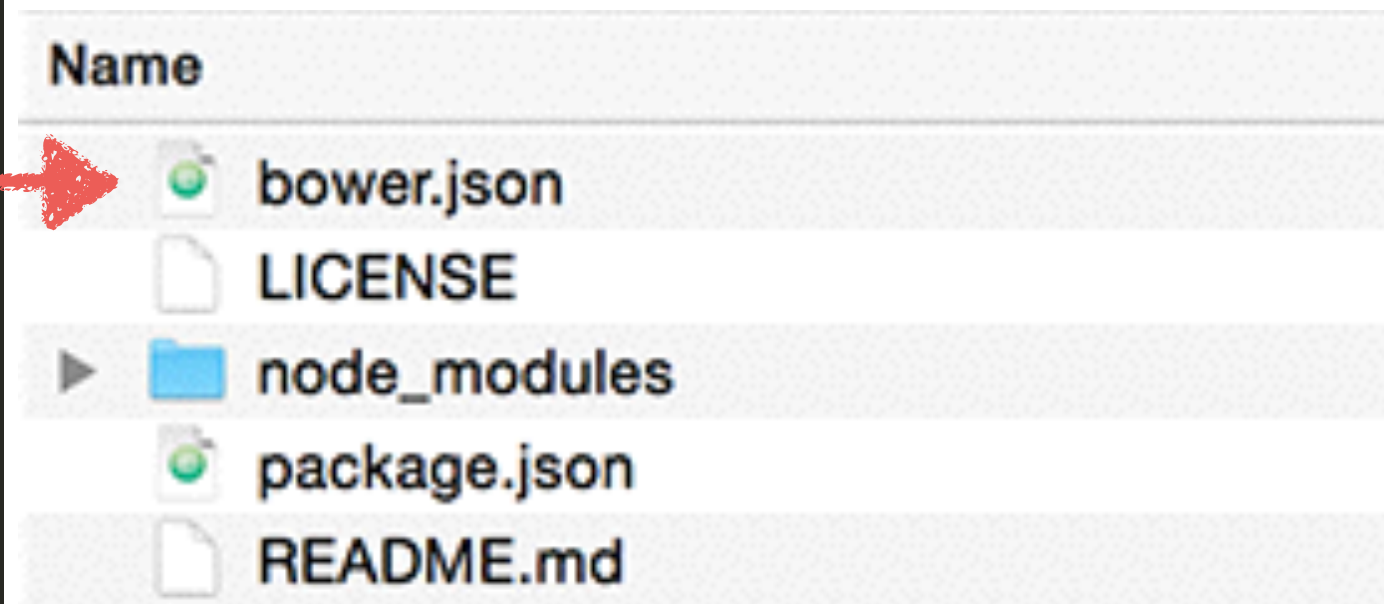


# The bower.json File

The bower.json file resides at the root level.

**\$ bower init**

```
{
  "name": "javascript-automation",
  "version": "1.0.0",
  "homepage": "https://github.com/craigphares/javascript-automation",
  "authors": [
    "Craig Phares <craigphares@sixoverground.com>"
  ],
  "license": "MIT",
  "ignore": [
    "**/*.*",
    "node_modules",
    "bower_components",
    "test",
    "tests"
  ]
}
```





# Let's Install jQuery

Bower components are stored in the bower\_components directory.

**\$ bower install jquery --save**

```
{
  "name": "javascript-automation",
  "version": "1.0.0",
  "homepage": "https://github.com/craigphares/javascript-automation",
  "authors": [
    "Craig Phares <craigphares@sixoverground.com>"
  ],
  "license": "MIT",
  "ignore": [
    "**/*.*",
    "node_modules",
    "bower_components",
    "test",
    "tests"
  ],
  "dependencies": {
    "jquery": "~2.1.4"
  }
}
```

Name	
▼	folder bower_components
→	folder jquery
	file bower.json
	file LICENSE
▶	folder node_modules
	file package.json
	file README.md

# Let's Install Some More

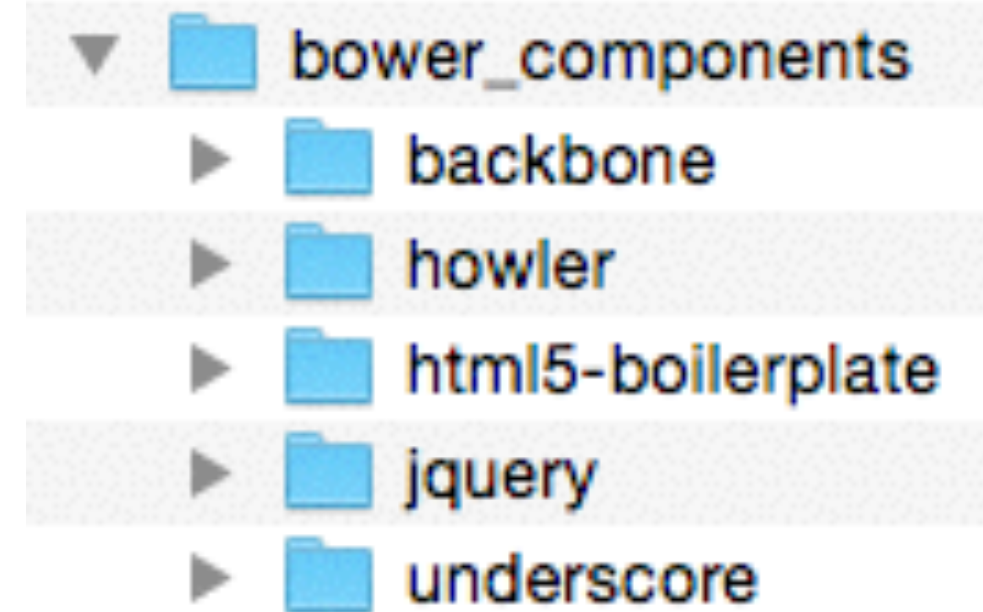
```
$ bower install html5-boilerplate --save
```

```
$ bower install underscore --save
```

```
$ bower install backbone --save
```

```
$ bower install howler --save
```

```
"dependencies": {  
  "jquery": "~2.1.4",  
  "html5-boilerplate": "~5.2.0",  
  "underscore": "~1.8.3",  
  "backbone": "~1.1.2",  
  "howler": "~1.1.26"  
}
```





# Test-Driven Development

Test-driven development (TDD) is a software development process that relies on the repetition of a very short development cycle:

Write a (initially failing) test case that defines a new feature

Produce the minimum amount of code to pass that test

Refactor the new code to acceptable standards

**TDD encourages simple designs and inspires confidence.**

Jasmine is a  
**JavaScript Testing Framework**



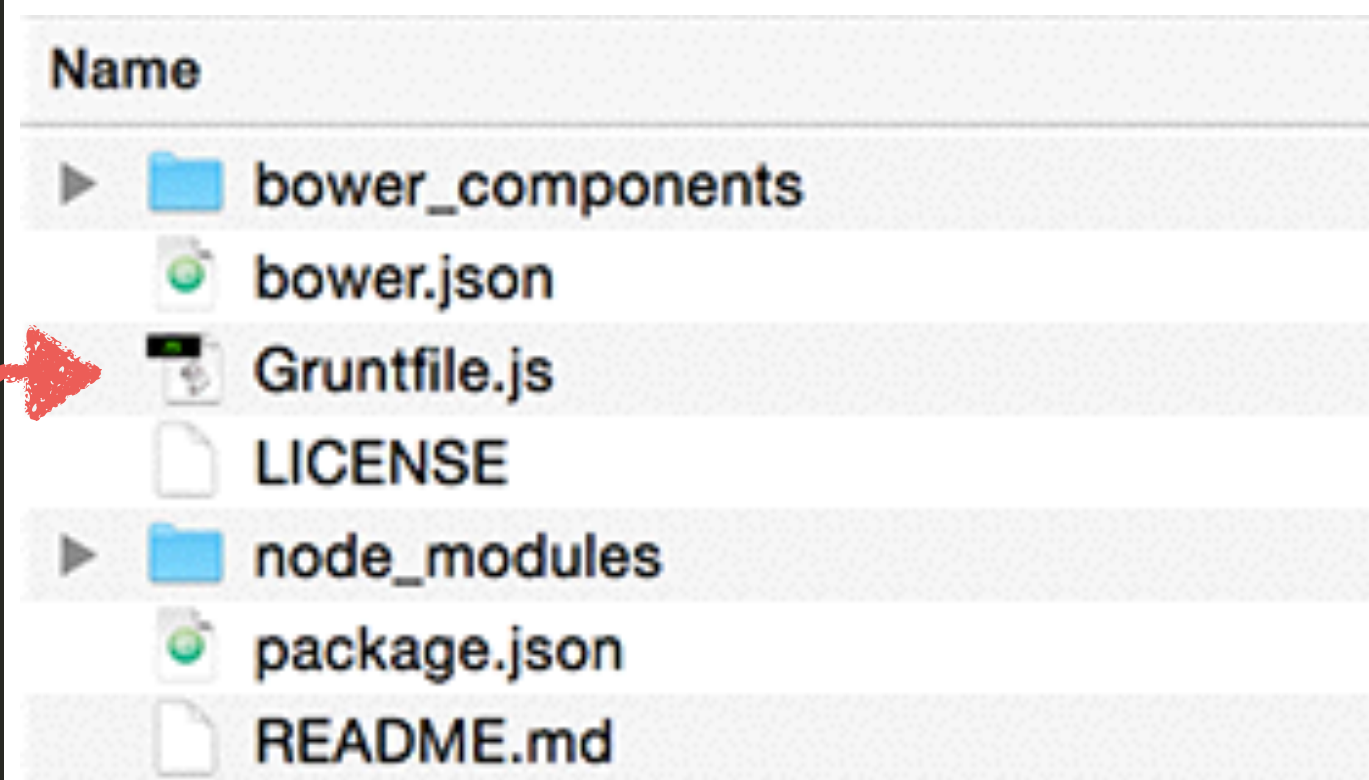
**Jasmine**

# The Gruntfile

The Gruntfile.js file resides at the root level.

```
$ npm install grunt --save-dev
```

```
'use strict';  
  
module.exports = function(grunt) {  
  
  grunt.initConfig({  
    pkg: grunt.file.readJSON('package.json'),  
  });  
  
};
```





# Let's install Jasmine

We're going to use Grunt to run Jasmine tests.

```
$ npm install grunt-contrib-jasmine --save-dev
```

We also need jasmine-query to handle some DOM manipulation.

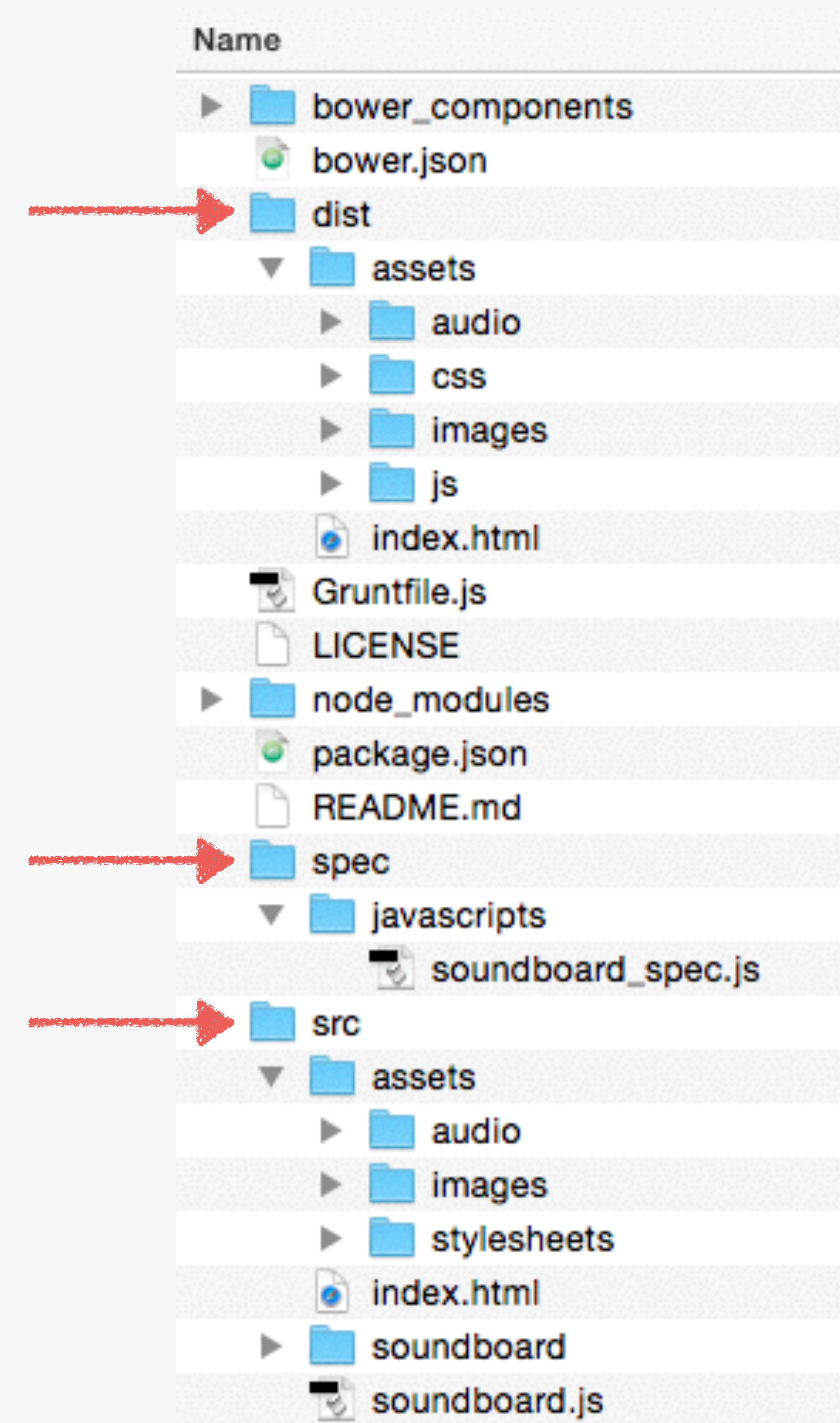
```
$ bower install jasmine-jquery --save-dev
```

# Folder Structure

Source belongs in /src

Specs belong in /spec

Production build resides in /dist





# Update the Gruntfile

```
module.exports = function(grunt) {  
  grunt.initConfig({  
    pkg: grunt.file.readJSON('package.json'),  
    jasmine: {  
      dist: {  
        src: [  
          'src/soundboard/soundboard.js',  
          'src/soundboard/models/**/*.js',  
          'src/soundboard/views/**/*.js',  
          'src/soundboard.js'  
        ],  
        options: {  
          specs: 'spec/**/*.spec.js',  
          helpers: 'spec/**/*.helper.js',  
          vendor: [  
            'bower_components/jquery/dist/jquery.js',  
            'bower_components/jasmine-jquery/lib/jasmine-jquery.js',  
            'bower_components/underscore/underscore.js',  
            'bower_components/backbone/backbone.js',  
            'bower_components/howler/howler.js'  
          ]  
        }  
      }  
    }  
  });  
};
```

Task

Target

Load the plugin

Register our task

```
grunt.loadNpmTasks('grunt-contrib-jasmine');
```

```
grunt.registerTask('test', ['jasmine']);
```

# Write a Test

Tests live in the spec folder.

We're going to load a collection of sounds for our project.

```
'use strict';

describe('Soundboard', function() {
  it('has sounds', function() {
    expect(SB.sounds).toBeDefined();
  });
});
```

# Run the Test

The test fails initially.

**\$ grunt test**

```
Craigs-MacBook-Pro:javascript-automation craigphares$ grunt test
Running "jasmine:dist" (jasmine) task
Testing jasmine specs via PhantomJS
```

## Soundboard

```
✗ has sounds
  Expected undefined to be defined. (1)
```

```
1 spec in 0.005s.
```

```
>> 1 failures
```

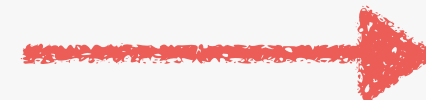
```
Warning: Task "jasmine:dist" failed. Use --force to continue.
```

```
Aborted due to warnings.
```



# Write Some Code

Name
▶ bower_components
• bower.json
• Gruntfile.js
• LICENSE
▶ node_modules
• package.json
• README.md
▼ spec
▼ javascripts
• soundboard_spec.js
▼ src
▶ assets
▼ soundboard
▼ models
• sound.js
• soundboard.js
• soundboard.js
▼ views
• sound.js
• soundboard.js
• starfield.js
• soundboard.js



# New Code

We're using Backbone to load a collection of sounds.

```
'use strict';

$(function() {

  SB.sounds = new SB.Collections.SoundCollection([
    { name: 'Base', url: 'assets/audio/base.mp3' },
    { name: 'Drums', url: 'assets/audio/drums.mp3' },
    { name: 'Loud Guitar', url: 'assets/audio/guitar-loud.mp3' },
    { name: 'Soft Guitar', url: 'assets/audio/guitar.mp3' },
    { name: 'Synth', url: 'assets/audio/synth.mp3' },
    { name: 'Voice', url: 'assets/audio/voice.mp3' },
    { name: 'Loud FX', url: 'assets/audio/zoops-loud.mp3' },
    { name: 'Soft FX', url: 'assets/audio/zoops.mp3' }
  ]);

});
```

# Run the Test

The test passes.

**\$ grunt test**

```
Craigs-MacBook-Pro:javascript-automation craighphares$ grunt test
Running "jasmine:dist" (jasmine) task
Testing jasmine specs via PhantomJS
```

**Soundboard**

✔ has sounds

1 spec in 0.004s.

>> 0 failures

Done, without errors.





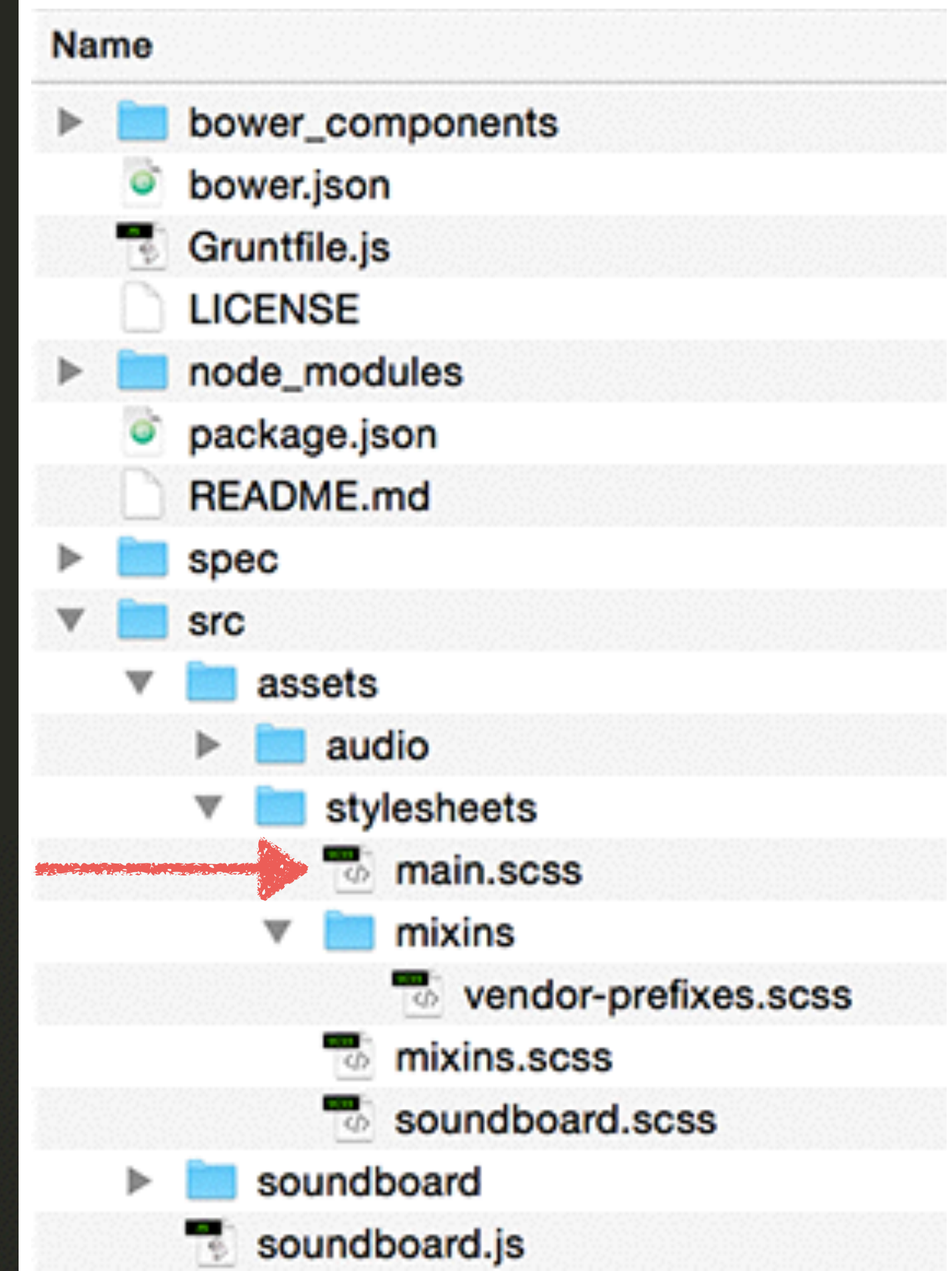
# Let's Add Some Sass

\$ npm install grunt-contrib-sass --save-dev  
And update the Gruntfile.

```
Task → sass: {  
  Target → dist: {  
    files: [{  
      expand: true,  
      cwd: 'src/assets/stylesheets/',  
      src: ['main.scss'],  
      dest: 'dist/assets/css',  
      ext: '.css'  
    }]  
  }  
};
```

```
Load the plugin → grunt.loadNpmTasks('grunt-contrib-jasmine');  
grunt.loadNpmTasks('grunt-contrib-sass');
```

```
Register our task → grunt.registerTask('default', ['sass']);
```

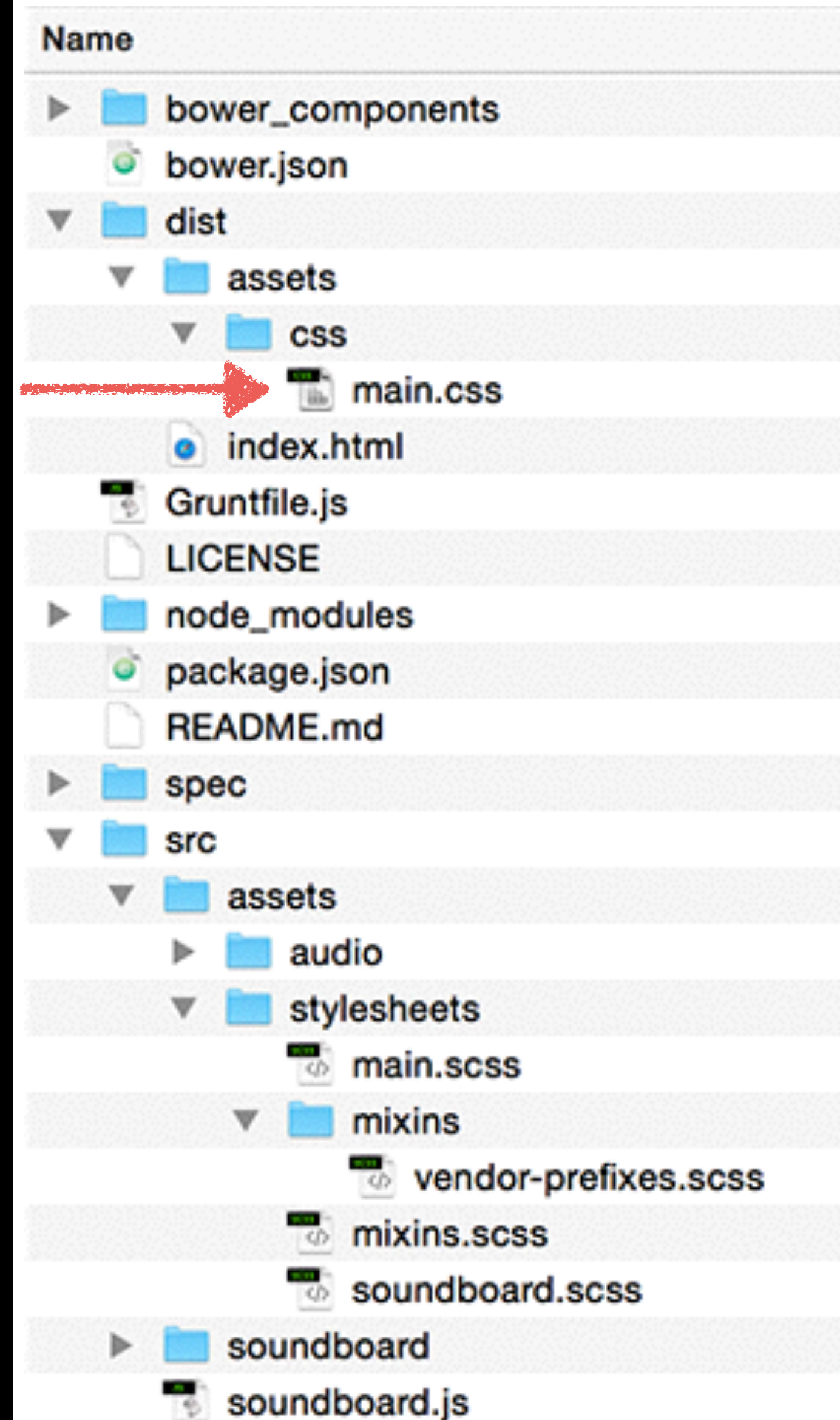


# Run The Default Task

\$ grunt

```
Craigs-MacBook-Pro:javascript-automation craigphares$ grunt  
Running "sass:dist" (sass) task
```

```
Done, without errors.
```





# Let's Improve This Build Script

Official Grunt plugins are named with "contrib".

```
$ npm install grunt-contrib-clean --save-dev  
$ npm install grunt-contrib-jshint --save-dev  
$ npm install grunt-contrib-concat --save-dev  
$ npm install grunt-contrib-uglify --save-dev  
$ npm install grunt-contrib-imagemin --save-dev  
$ npm install grunt-contrib-copy --save-dev
```

# Clean

```
clean: {  
  dist: ['dist']  
},
```

# JSHint

```
jshint: {  
  all: ['Gruntfile.js', 'src/**/*.js', 'spec/**/*.js'],  
  options: {  
    jshinttrc: '.jshinttrc'  
  }  
},
```

# Concat

```
concat: {
  css: {
    src: [
      'bower_components/html5-boilerplate/dist/css/normalize.css',
      'bower_components/html5-boilerplate/dist/css/main.css',
      'dist/assets/css/main.css'
    ],
    dest: 'dist/assets/css/app.css'
  },
  js: {
    src: [
      'bower_components/jquery/dist/jquery.js',
      'bower_components/howler/howler.js',
      'bower_components/underscore/underscore.js',
      'bower_components/backbone/backbone.js',
      'src/soundboard/soundboard.js',
      'src/soundboard/models/**/*.js',
      'src/soundboard/views/**/*.js',
      'src/soundboard.js'
    ],
    dest: 'dist/assets/js/app.js'
  }
},
```



# Uglify

```
uglify: {  
  dist: {  
    files: {  
      'dist/assets/js/app.min.js': ['dist/assets/js/app.js']  
    }  
  }  
},
```

# Imagemin

```
imagemin: {  
  dist: {  
    files: [{  
      expand: true,  
      cwd: 'src/assets/images/',  
      src: ['**/*.{png,jpg,gif}'],  
      dest: 'dist/assets/images/'  
    }]  
  }  
},
```



# Copy

```
copy: {
  audio: {
    files: [
      {
        expand: true,
        cwd: 'src/assets/audio/',
        src: ['**/*'],
        dest: 'dist/assets/audio/'
      }
    ]
  },
  html: {
    files: [
      {
        expand: true,
        cwd: 'src/',
        src: ['**/*.html'],
        dest: 'dist/'
      }
    ]
  }
},
```




# The New Gruntfile

Build the application with one command.

**\$ grunt**

```
grunt.loadNpmTasks('grunt-contrib-jasmine');
grunt.loadNpmTasks('grunt-contrib-sass');
grunt.loadNpmTasks('grunt-contrib-clean');
grunt.loadNpmTasks('grunt-contrib-jshint');
grunt.loadNpmTasks('grunt-contrib-concat');
grunt.loadNpmTasks('grunt-contrib-uglify');
grunt.loadNpmTasks('grunt-contrib-imagemin');
grunt.loadNpmTasks('grunt-contrib-copy');

grunt.registerTask('test', ['jasmine']);


grunt.registerTask('default', ['clean', 'sass', 'jshint', 'concat', 'uglify',
    ', 'copy']);
```

```
Craigs-MacBook-Pro:javascript-automation craigphares$ grunt
Running "clean:dist" (clean) task
>> 1 path cleaned.
```

```
Running "sass:dist" (sass) task
```

```
Running "jshint:all" (jshint) task
>> 9 files lint free.
```

```
Running "concat:css" (concat) task
File dist/assets/css/app.css created.
```

```
Running "concat:js" (concat) task
File dist/assets/js/app.js created.
```

```
Running "uglify:dist" (uglify) task
>> 1 file created.
```

```
Running "imagemin:dist" (imagemin) task
Minified 3 images (saved 30.71 kB)
```

```
Running "copy:audio" (copy) task
Copied 9 files
```

```
Running "copy:html" (copy) task
Copied 1 file
```

```
Done, without errors.
```



# Our Build

Name
▶ folder bower_components
📄 bower.json
▶ folder dist
▼ folder assets
▶ folder audio
▼ folder css
📄 app.css
📄 main.css
▶ folder images
▼ folder js
📄 app.js
📄 app.min.js
📄 index.html
📄 Gruntfile.js
📄 LICENSE
▶ folder node_modules
📄 package.json
📄 README.md
▶ folder spec
▶ folder src



# Need a Web Server?

Use connect to boot a server wherever you like.

\$ npm install grunt-contrib-connect --save-dev

```
connect: {  
  server: {  
    options: {  
      port: 8000,  
      base: 'dist'  
    }  
  }  
};  
  
grunt.loadNpmTasks('grunt-contrib-jasmine');  
grunt.loadNpmTasks('grunt-contrib-sass');  
grunt.loadNpmTasks('grunt-contrib-clean');  
grunt.loadNpmTasks('grunt-contrib-jshint');  
grunt.loadNpmTasks('grunt-contrib-concat');  
grunt.loadNpmTasks('grunt-contrib-uglify');  
grunt.loadNpmTasks('grunt-contrib-imagemin');  
grunt.loadNpmTasks('grunt-contrib-copy');  
grunt.loadNpmTasks('grunt-contrib-watch');  
grunt.loadNpmTasks('grunt-contrib-connect');  
  
grunt.registerTask('start', ['connect:server:keepalive']);
```

\$ grunt start

```
Craigs-MacBook-Pro:javascript-automation craighphares$ grunt start  
Running "connect:server:keepalive" (connect) task  
Waiting forever...  
Started connect web server on http://localhost:8000
```



# Let's Automate

Use watch to selectively watch your source for changes.

```
$ npm install grunt-contrib-watch --save-dev
```

```
watch: {
  options: {
    livereload: true
  },
  css: {
    files: ['src/**/*.scss'],
    tasks: ['sass', 'concat:css'],
    options: {
      spawn: false
    }
  },
  js: {
    files: ['src/**/*.js'],
    tasks: ['jshint', 'concat:js', 'uglify'],
    options: {
      spawn: false
    }
  },
  html: {
    files: ['src/**/*.html'],
    tasks: ['copy:html'],
    options: {
      spawn: false
    }
  }
}
```

## \$ grunt watch

```
Craigs-MacBook-Pro:javascript-automation craighares$ grunt watch
Running "watch" task
Waiting...
>> File "src/assets/stylesheets/soundboard.scss" changed.

Running "sass:dist" (sass) task

Running "concat:css" (concat) task
File dist/assets/css/app.css created.

Running "watch" task
Completed in 0.499s at Tue May 12 2015 19:56:57 GMT-0400 (EDT) - Waiting...
```

Use LiveReload to automatically refresh the browser.



LiveReload 2.1.0  
[Permissions](#) [Details](#)

Enabled



Allow in incognito  Allow access to file URLs



# Templating

A little known feature of Grunt is that it can process Lo-Dash templates.

index.html

```
<div class="bobblehead <%= bobblehead %>">
  <div class="base"></div>
  <div class="head"></div>
</div>
```

Gruntfile.js

```
module.exports = function(grunt) {

  var bobblehead = grunt.option('bobblehead') || 'stormtrooper';

  grunt.initConfig({
    pkg: grunt.file.readJSON('package.json'),

    bobblehead: bobblehead,

    copy: {
      audio: {
        files: [
          {
            expand: true,
            cwd: 'src/assets/audio/',
            src: ['**/*'],
            dest: 'dist/assets/audio/'
          }
        ]
      },
      index: {
        src: 'src/index.html',
        dest: 'dist/index.html',
        options: {
          process: function(content, srcpath) {
            return grunt.template.process(content);
          }
        }
      }
    }
  });
};
```

# Command Line Arguments

All Grunt command line arguments are passed to the `grunt.options` object.

**\$ grunt --bobblehead=boba-fett**

```
Craigs-MacBook-Pro:javascript-automation craigphares$ grunt --bobblehead=boba-fett
Running "clean:dist" (clean) task
>> 1 path cleaned.

Running "sass:dist" (sass) task

Running "jshint:all" (jshint) task
>> 9 files lint free.

Running "concat:css" (concat) task
File dist/assets/css/app.css created.

Running "concat:js" (concat) task
File dist/assets/js/app.js created.

Running "uglify:dist" (uglify) task
>> 1 file created.

Running "imagemin:dist" (imagemin) task
Minified 5 images (saved 35.45 kB)

Running "copy:audio" (copy) task
Copied 10 files

Running "copy:index" (copy) task
Copied 1 file

Done, without errors.
```

# The Possibilities Are Endless

Compile CoffeeScript with **grunt-contrib-coffee**

Minimize CSS with **grunt-contrib-cssmin**

Include more complex templates using **grunt-processhtml**

Upload your build to a server using **grunt-ftp-deploy**

Run shell commands with **grunt-shell**

Visit [gruntjs.com/plugins](http://gruntjs.com/plugins) for a full list of Grunt plugins



# Stop Doing Mundane Work



@craigphares

[craigphares.github.io/javascript-automation/](https://craigphares.github.io/javascript-automation/)