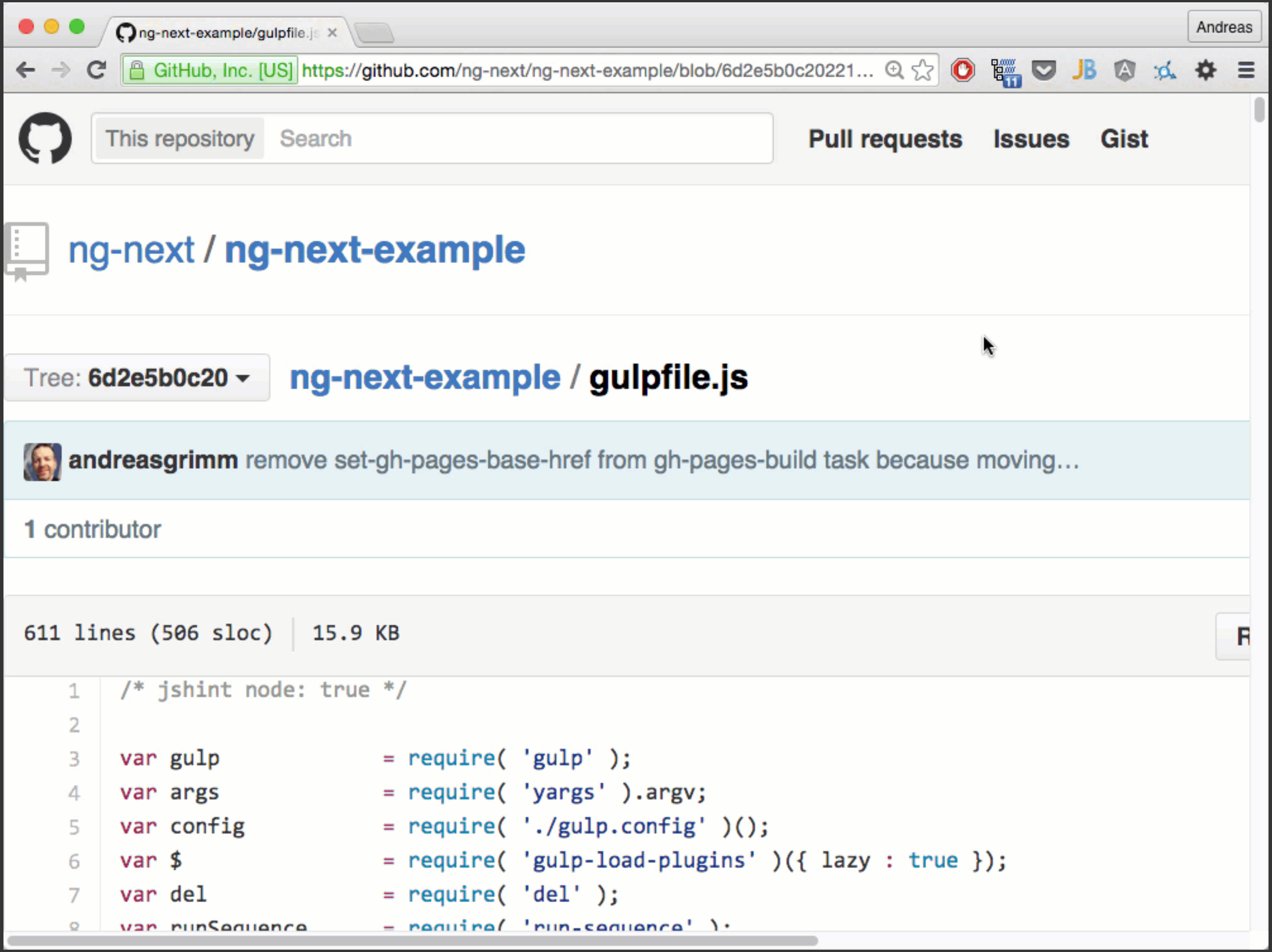
Clean JavaScript

my experiences writing Clean Code in the language Uncle Bob doesn’t talk about (too much)

Dirty JavaScript

we all have seen it

Big Files


ng-next-example/gulpfile.js
Andreas
GitHub, Inc. [US]
https://github.com/ng-next/ng-next-example/blob/6d2e5b0c20221…
This repository
Search
Pull requests
Issues
Gist
ng-next / ng-next-example
Tree: 6d2e5b0c20
ng-next-example / gulpfile.js
andreasgrimm remove set-gh-pages-base-href from gh-pages-build task because moving…
1 contributor
611 lines (506 sloc) | 15.9 KB
R

```
/* jshint node: true */

var gulp                = require( 'gulp' );
var args                = require( 'yargs' ).argv;
var config              = require( './gulp.config' )();
var $                   = require( 'gulp-load-plugins' )({ lazy : true });
var del                 = require( 'del' );
var runSequence         = require( 'run-sequence' );
```

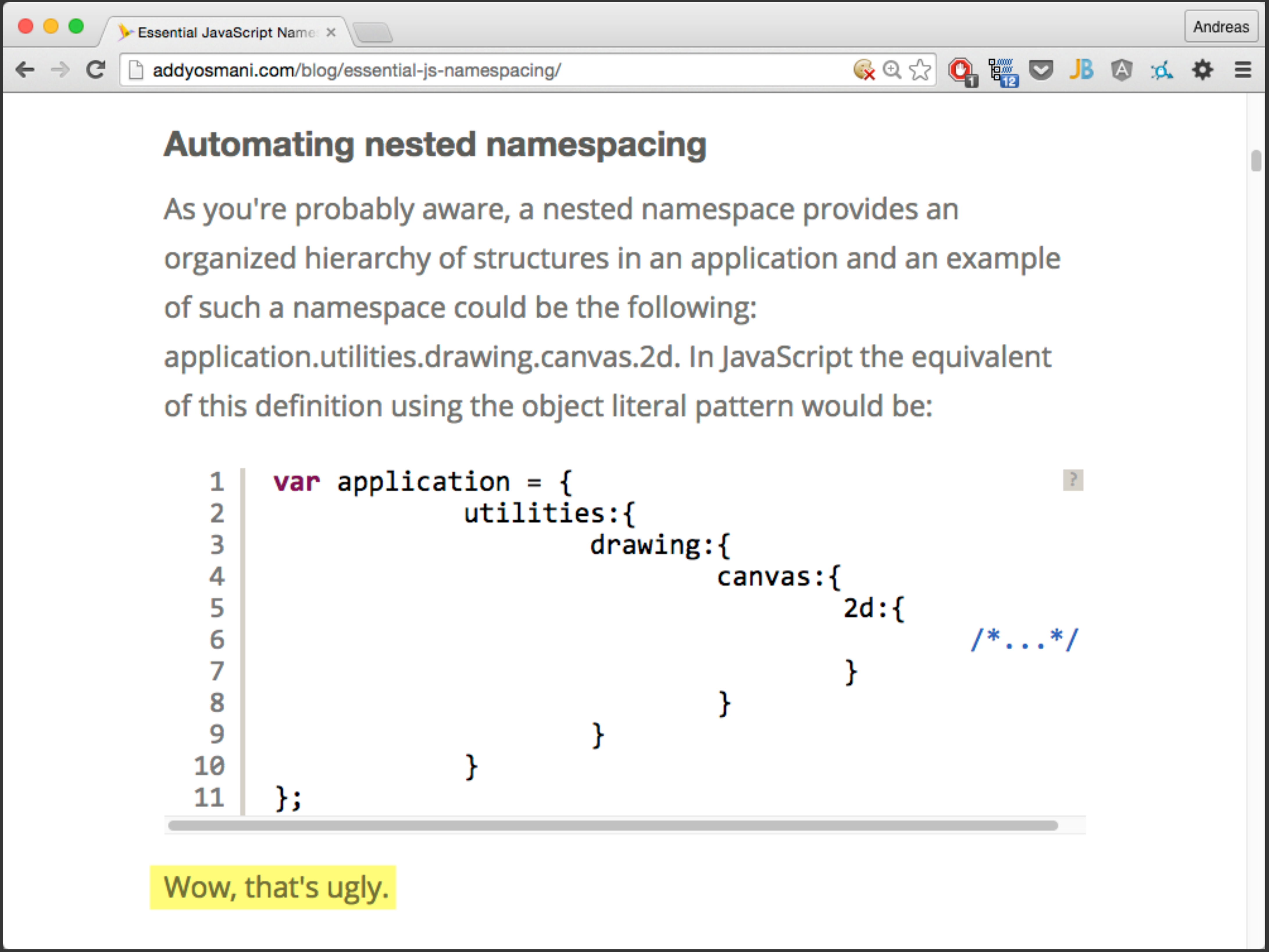
One Single **Big** File

Big Methods

this = that

this = self

No Namespace Concept, Only Globals



Essential JavaScript Name ×
Andreas
addyosmani.com/blog/essential-js-namespacing/

Automating nested namespacing

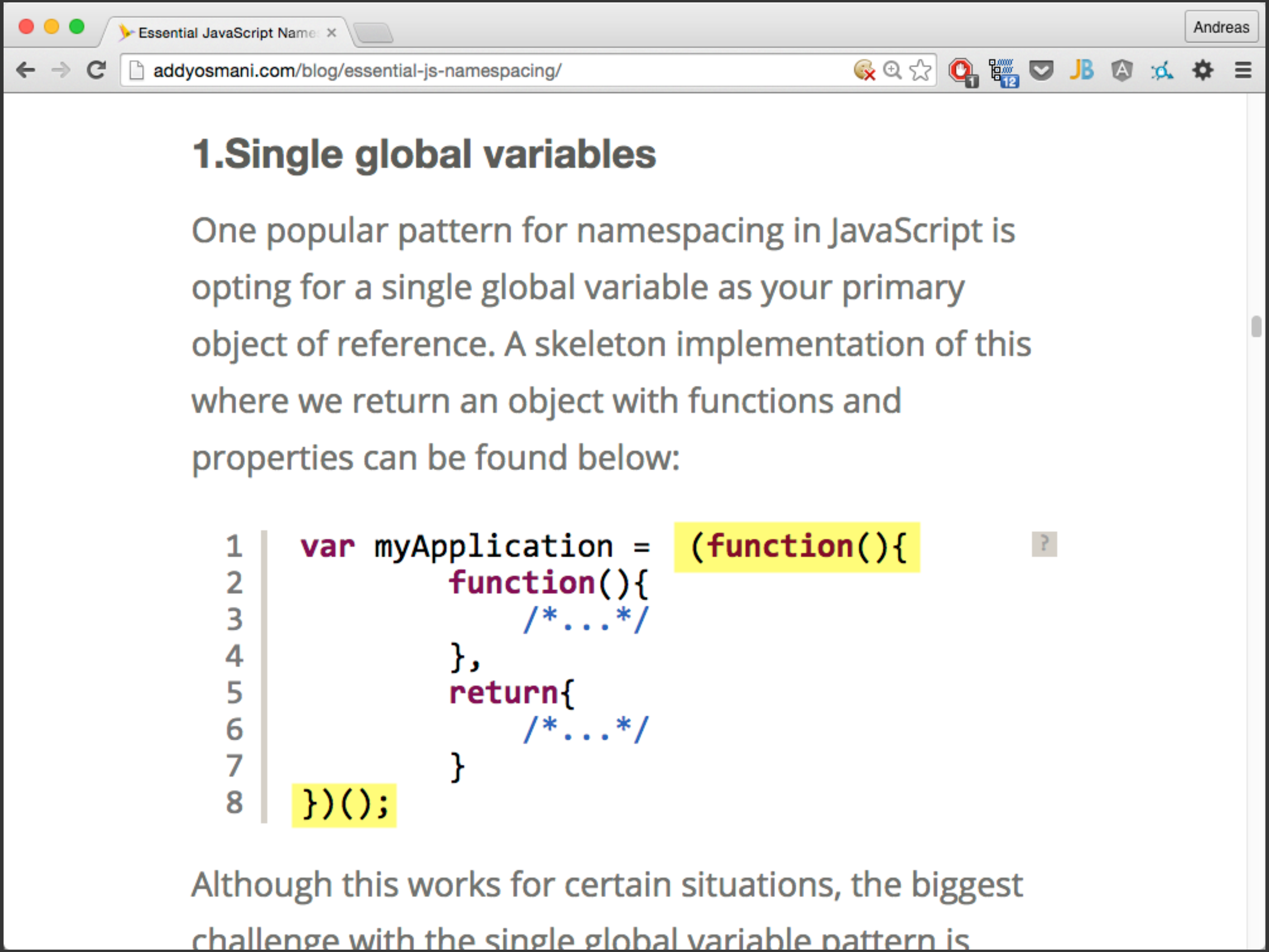
As you're probably aware, a nested namespace provides an organized hierarchy of structures in an application and an example of such a namespace could be the following: application.utilities.drawing.canvas.2d. In JavaScript the equivalent of this definition using the object literal pattern would be:

```
var application = {
            utilities:{
                    drawing:{
                            canvas:{
                                    2d:{
                                            /*...*/
                                    }
                            }
                    }
            }
};
```

Wow, that's ugly.

IIFE

Immediately-invoked function expression



Essential JavaScript Name ×
addyosmani.com/blog/essential-js-namespacing/
Andreas

1.Single global variables

One popular pattern for namespacing in JavaScript is opting for a single global variable as your primary object of reference. A skeleton implementation of this where we return an object with functions and properties can be found below:

```
var myApplication =  (function(){
        function(){
            /*...*/
        },
        return{
            /*...*/
        }
})();
```

Although this works for certain situations, the biggest challenge with the single global variable pattern is

Bad JavaScript Parts

with

void

eval

type coercion

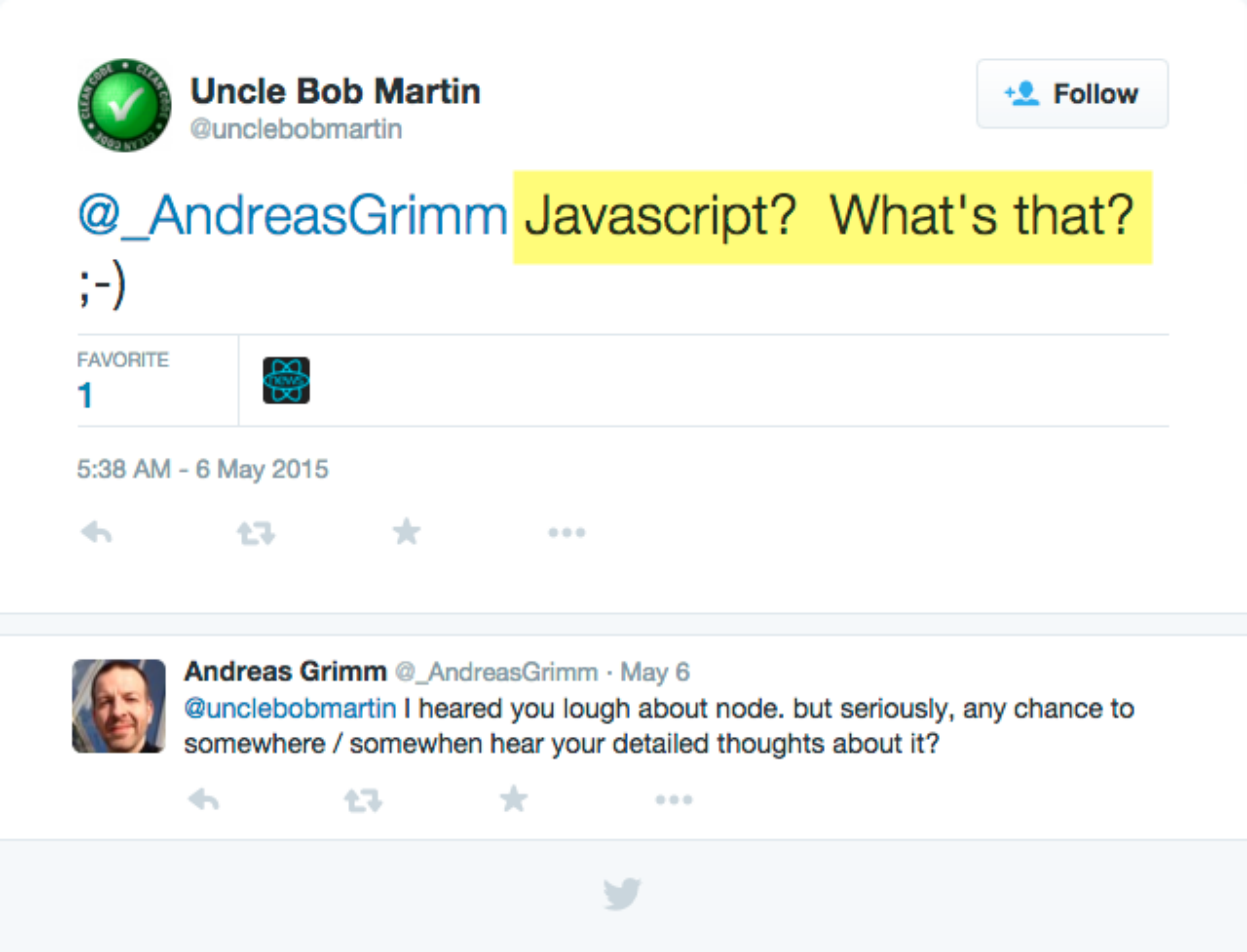
continue/switch

ES5 sloppy mode

Clean JavaScript

Is it possible?

Ask Uncle Bob!



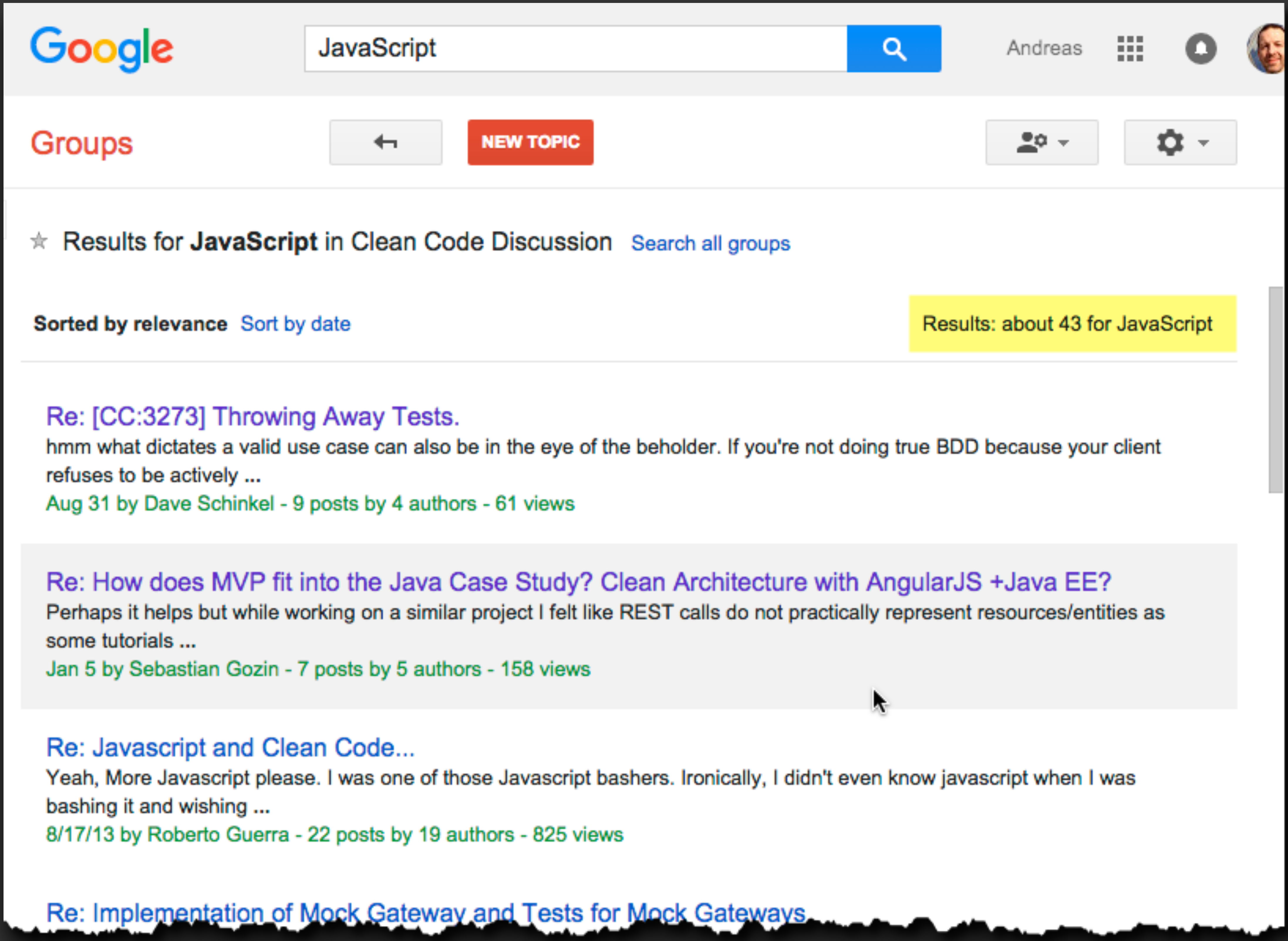
Uncle Bob Martin
@unclebobmartin

@_AndreasGrimm Javascript? What's that? ;-)

FAVORITE
1

5:38 AM - 6 May 2015

Andreas Grimm @_AndreasGrimm · May 6
@unclebobmartin I heared you lough about node. but seriously, any chance to somewhere / somewhen hear your detailed thoughts about it?



☆ Results for **JavaScript** in Clean Code Discussion Search all groups

Sorted by relevance Sort by date

Results: about 43 for JavaScript

Re: [CC:3273] Throwing Away Tests.
hmm what dictates a valid use case can also be in the eye of the beholder. If you're not doing true BDD because your client refuses to be actively ...
Aug 31 by Dave Schinkel - 9 posts by 4 authors - 61 views

Re: How does MVP fit into the Java Case Study? Clean Architecture with AngularJS +Java EE?
Perhaps it helps but while working on a similar project I felt like REST calls do not practically represent resources/entities as some tutorials ...
Jan 5 by Sebastian Gozin - 7 posts by 5 authors - 158 views

Re: Javascript and Clean Code...
Yeah, More Javascript please. I was one of those Javascript bashers. Ironically, I didn't even know javascript when I was bashing it and wishing ...
8/17/13 by Roberto Guerra - 22 posts by 19 authors - 825 views

Re: Implementation of Mock Gateway and Tests for Mock Gateways

groups.google.com/forum/?hl=en#!searchin/clean-code-discussion/Jav

JavaScript can be Clean

JavaScript can be Clean(er)

ES5 "use strict"; mode

Browser-native

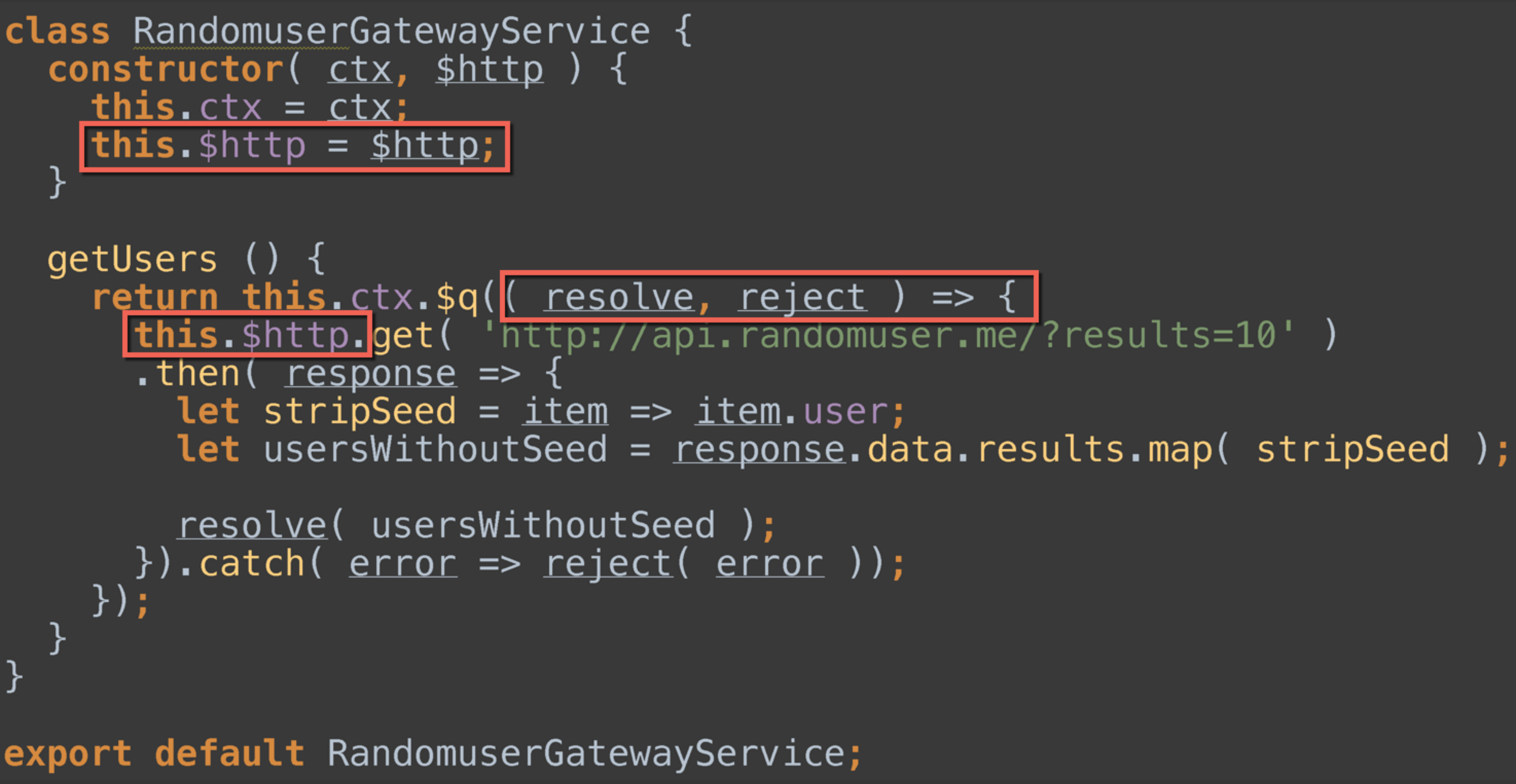
ES6

ES2015

ES7

ES2016

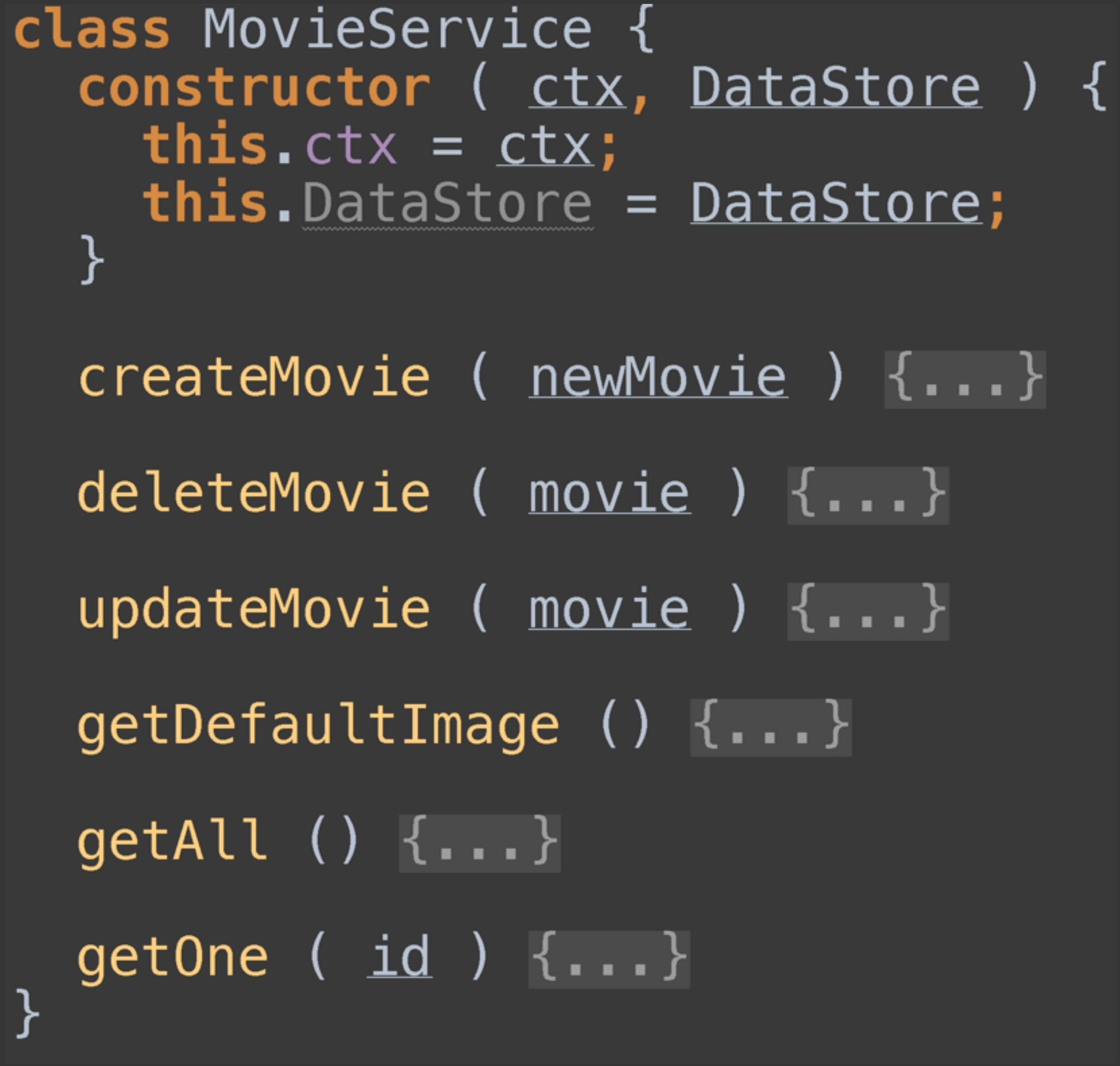
Fat Arrow Functions



```
class RandomuserGatewayService {
  constructor( ctx, $http ) {
    this.ctx = ctx;
    this.$http = $http;
  }

  getUsers () {
    return this.ctx.$q(( resolve, reject ) => {
      this.$http.get( 'http://api.randomuser.me/?results=10' )
      .then( response => {
        let stripSeed = item => item.user;
        let usersWithoutSeed = response.data.results.map( stripSeed );

        resolve( usersWithoutSeed );
      }).catch( error => reject( error ));
    });
  }
}

export default RandomuserGatewayService;
```

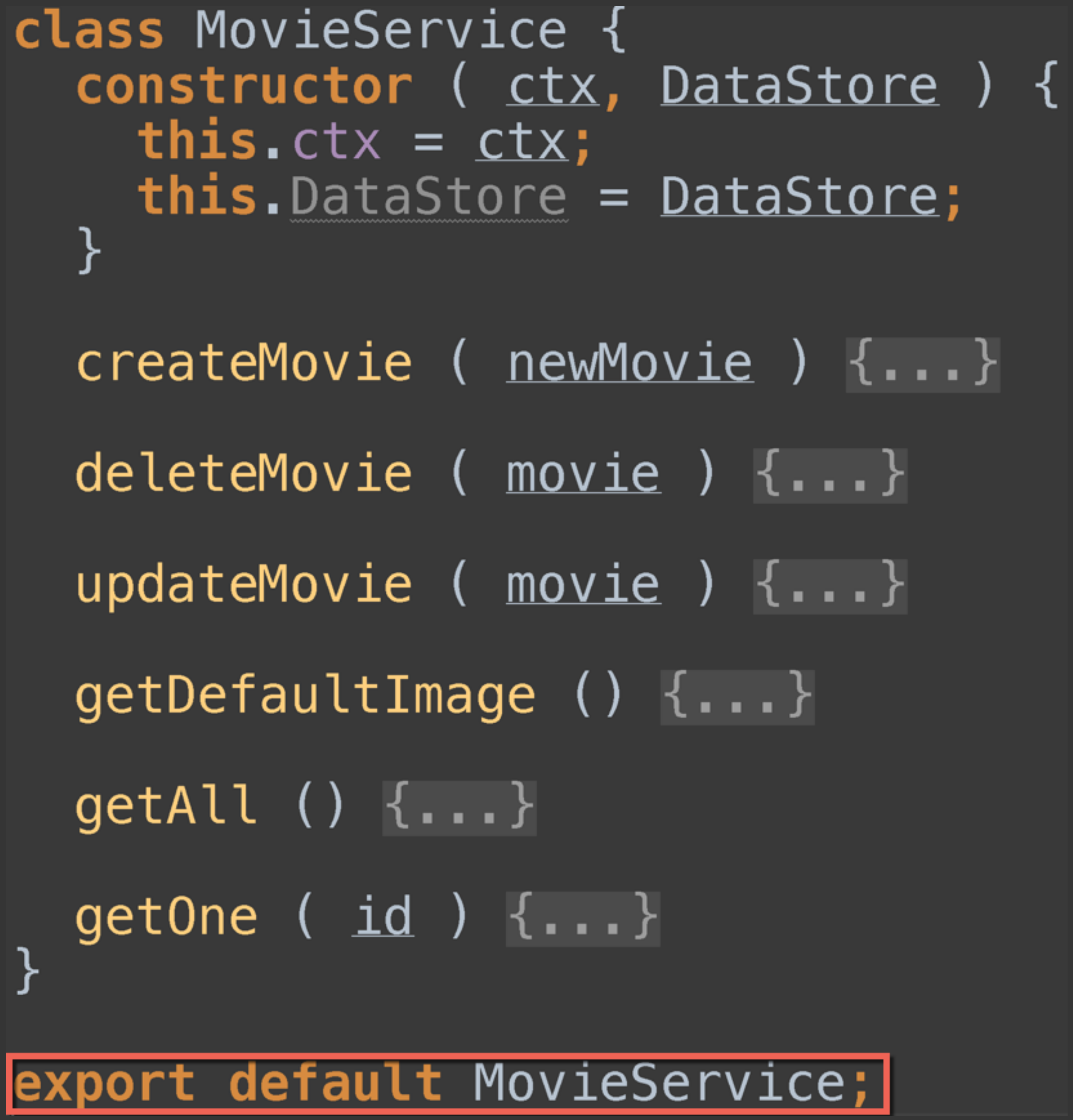
this = this

Classes



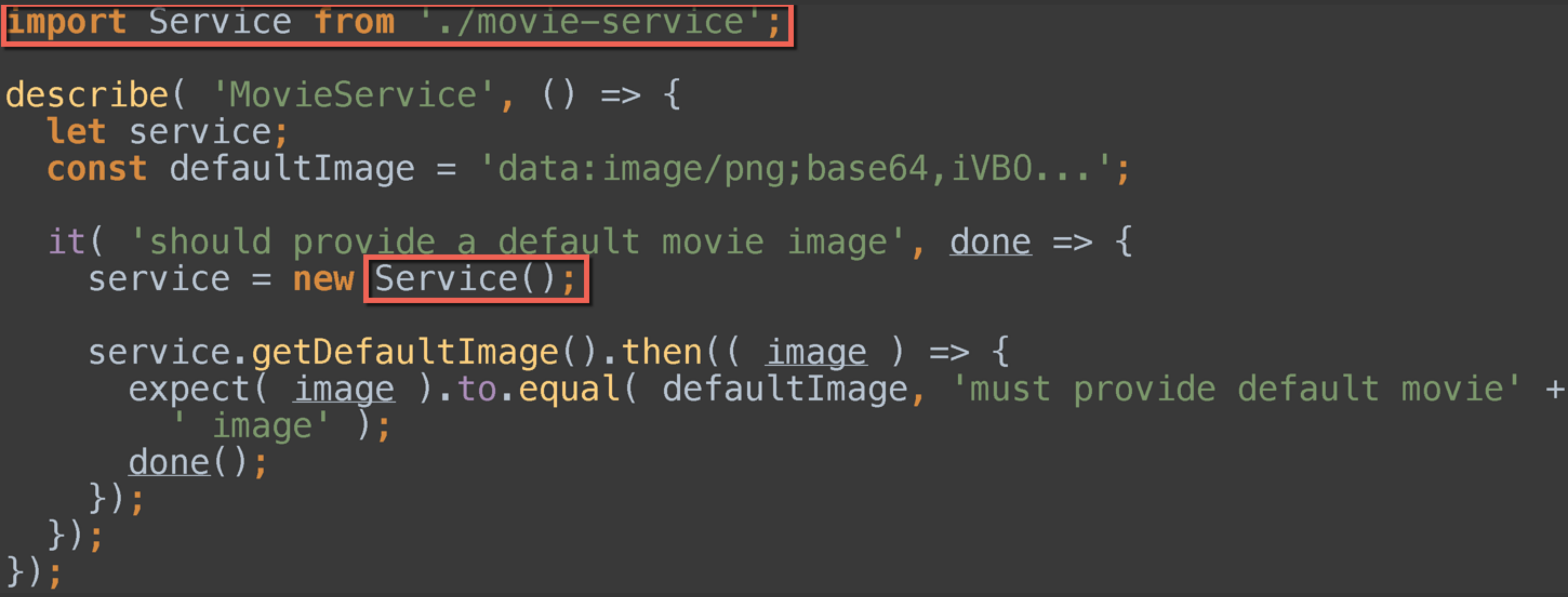
```
class MovieService {
  constructor ( ctx, DataStore ) {
    this.ctx = ctx;
    this.DataStore = DataStore;
  }

  createMovie ( newMovie ) {...}

  deleteMovie ( movie ) {...}

  updateMovie ( movie ) {...}

  getDefaultImage () {...}

  getAll () {...}

  getOne ( id ) {...}
}
```

(ES6) Modules



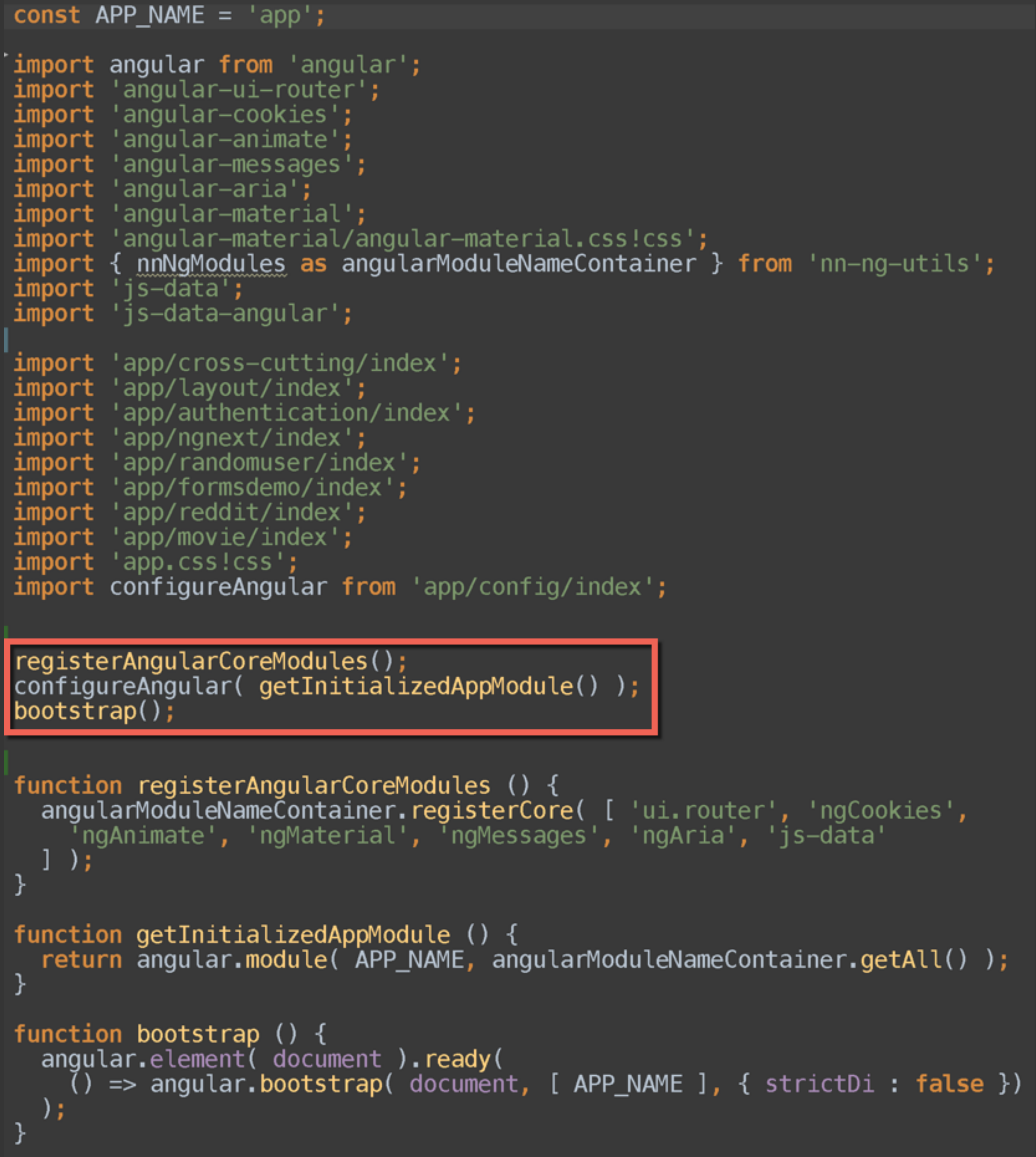
```
class MovieService {
  constructor ( ctx, DataStore ) {
    this.ctx = ctx;
    this.DataStore = DataStore;
  }

  createMovie ( newMovie ) {...}

  deleteMovie ( movie ) {...}

  updateMovie ( movie ) {...}

  getDefaultImage () {...}

  getAll () {...}

  getOne ( id ) {...}
}

export default MovieService;
```

(ES6) Modules



```
import Service from './movie-service';

describe( 'MovieService', () => {
  let service;
  const defaultImage = 'data:image/png;base64,iVBO...';

  it( 'should provide a default movie image', done => {
    service = new Service();

    service.getDefaultImage().then(( image ) => {
      expect( image ).to.equal( defaultImage, 'must provide default movie' +
        ' image' );
      done();
    });
  });
});
```

main.js

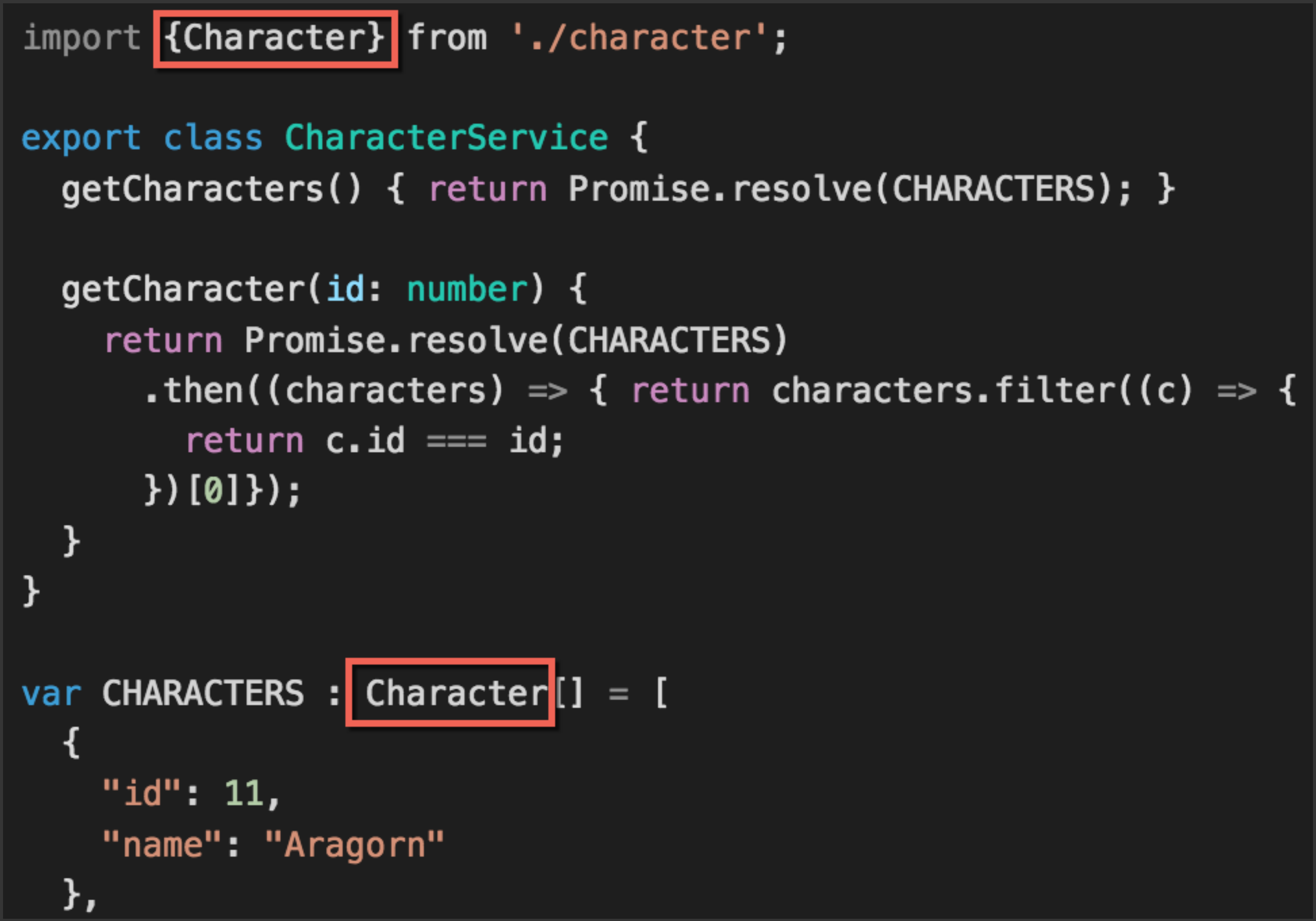


```
const APP_NAME = 'app';

import angular from 'angular';
import 'angular-ui-router';
import 'angular-cookies';
import 'angular-animate';
import 'angular-messages';
import 'angular-aria';
import 'angular-material';
import 'angular-material/angular-material.css!css';
import { nnNgModules as angularModuleNameContainer } from 'nn-ng-utils';
import 'js-data';
import 'js-data-angular';

import 'app/cross-cutting/index';
import 'app/layout/index';
import 'app/authentication/index';
import 'app/ngnext/index';
import 'app/randomuser/index';
import 'app/formsdemo/index';
import 'app/reddit/index';
import 'app/movie/index';
import 'app.css!css';
import configureAngular from 'app/config/index';

registerAngularCoreModules();
configureAngular( getInitializedAppModule() );
bootstrap();

function registerAngularCoreModules () {
  angularModuleNameContainer.registerCore( [ 'ui.router', 'ngCookies',
    'ngAnimate', 'ngMaterial', 'ngMessages', 'ngAria', 'js-data'
  ] );
}

function getInitializedAppModule () {
  return angular.module( APP_NAME, angularModuleNameContainer.getAll() );
}

function bootstrap () {
  angular.element( document ).ready(
    () => angular.bootstrap( document, [ APP_NAME ], { strictDi : false })
  );
}
```

TypeScript

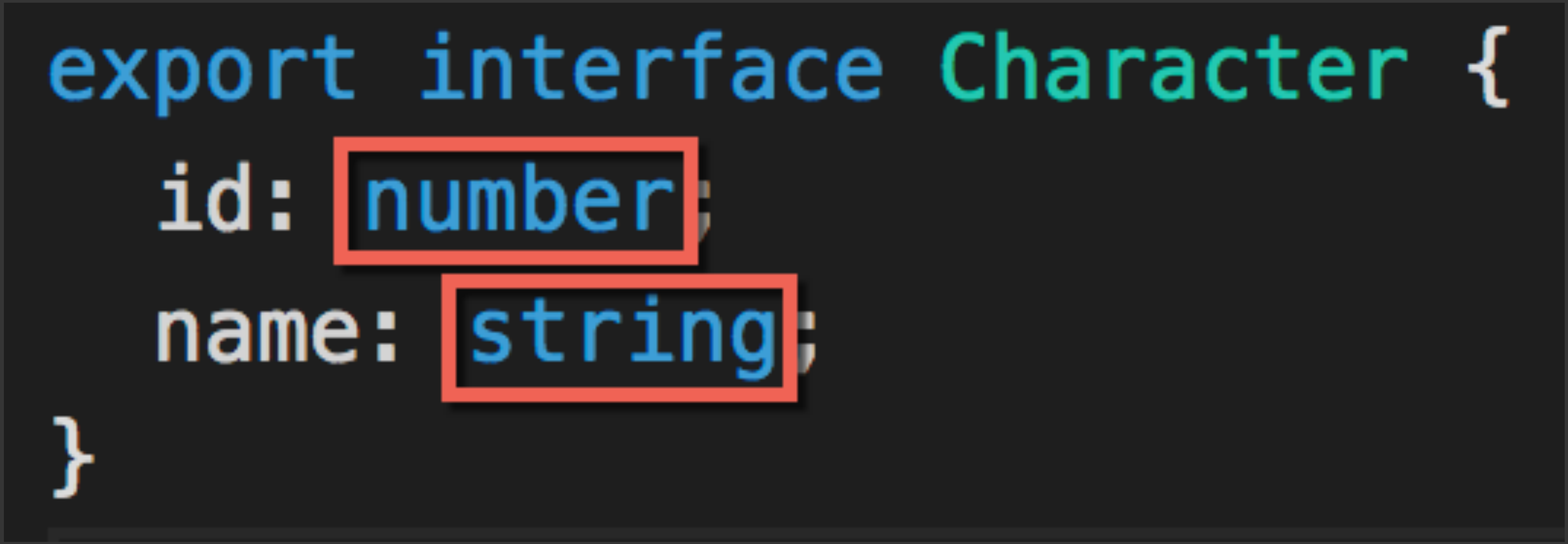
superset of JavaScript (ES6)

Types



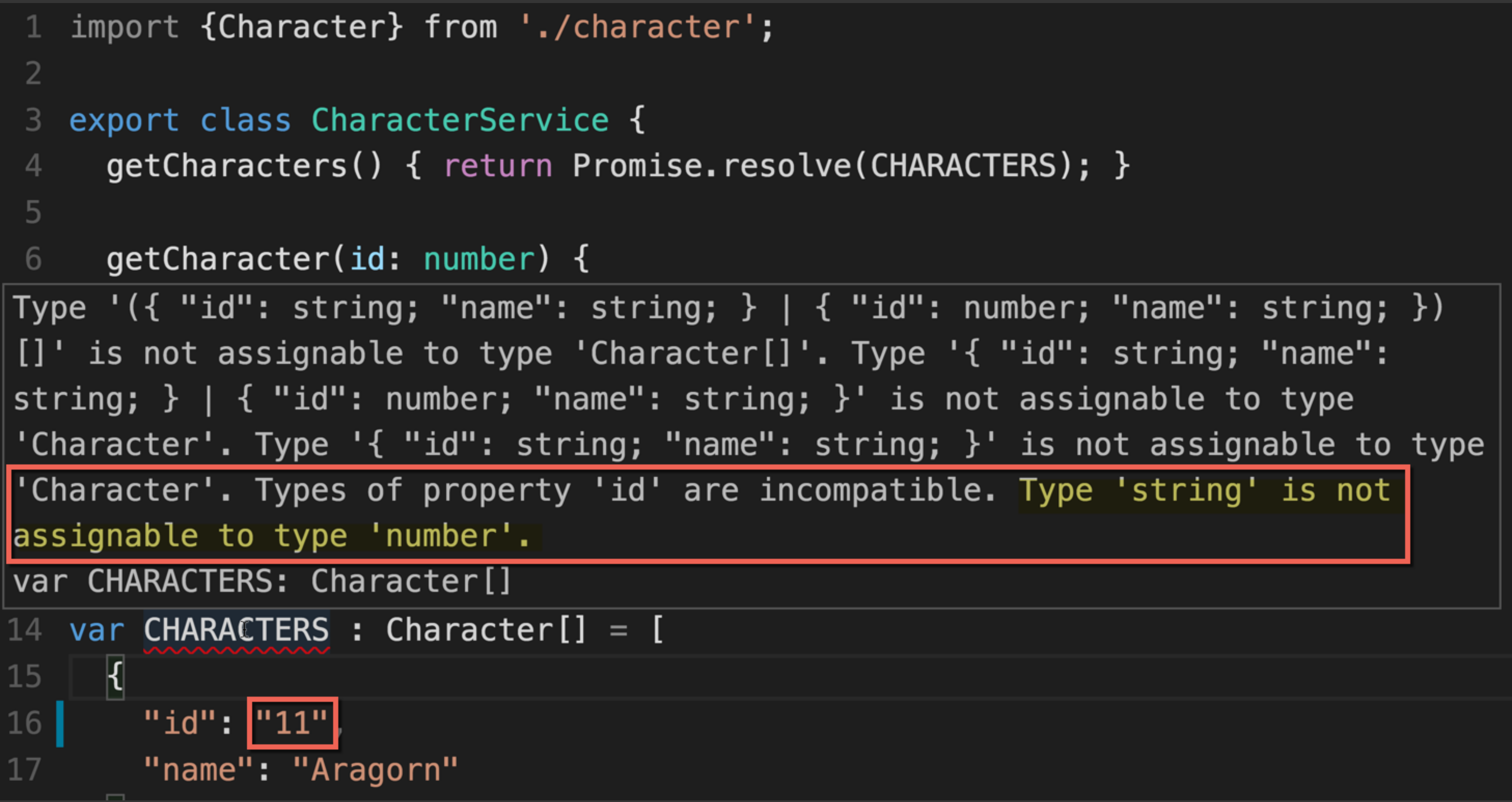
```
import {Character} from './character';

export class CharacterService {
  getCharacters() { return Promise.resolve(CHARACTERS); }

  getCharacter(id: number) {
    return Promise.resolve(CHARACTERS)
      .then((characters) => { return characters.filter((c) => {
        return c.id === id;
      })[0]});
  }
}

var CHARACTERS : Character[] = [
  {
    "id": 11,
    "name": "Aragorn"
  },
```

Types



```
export interface Character {
  id: number;
  name: string;
}
```

Types

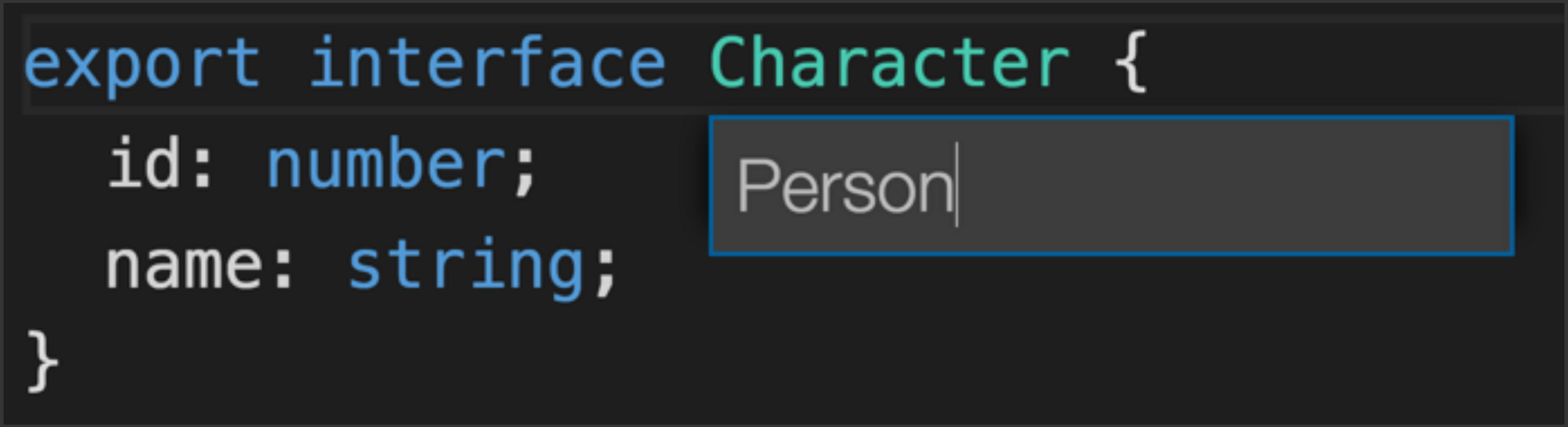


```
import {Character} from './character';

export class CharacterService {
  getCharacters() { return Promise.resolve(CHARACTERS); }

  getCharacter(id: number) {
```

```
Type '({ "id": string; "name": string; } | { "id": number; "name": string; })
[]' is not assignable to type 'Character[]'. Type '{ "id": string; "name":
string; } | { "id": number; "name": string; }' is not assignable to type
'Character'. Type '{ "id": string; "name": string; }' is not assignable to type
'Character'. Types of property 'id' are incompatible. Type 'string' is not
assignable to type 'number'.
var CHARACTERS: Character[]
```

```
var CHARACTERS : Character[] = [
  {
    "id": "11",
    "name": "Aragorn"
```

IDE Tooling

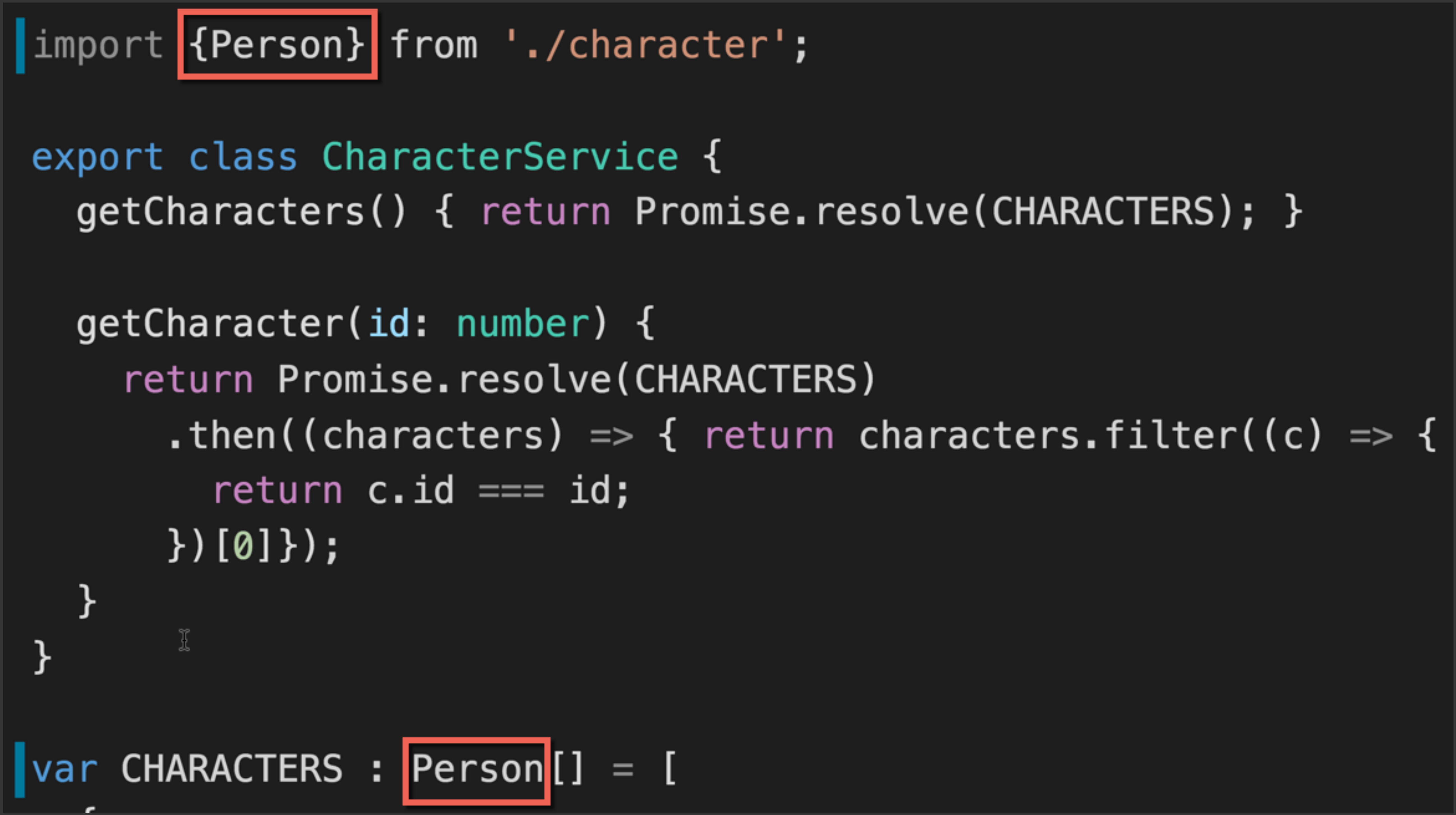


```
export interface Character {
  id: number;
  name: string;
}
```

Person

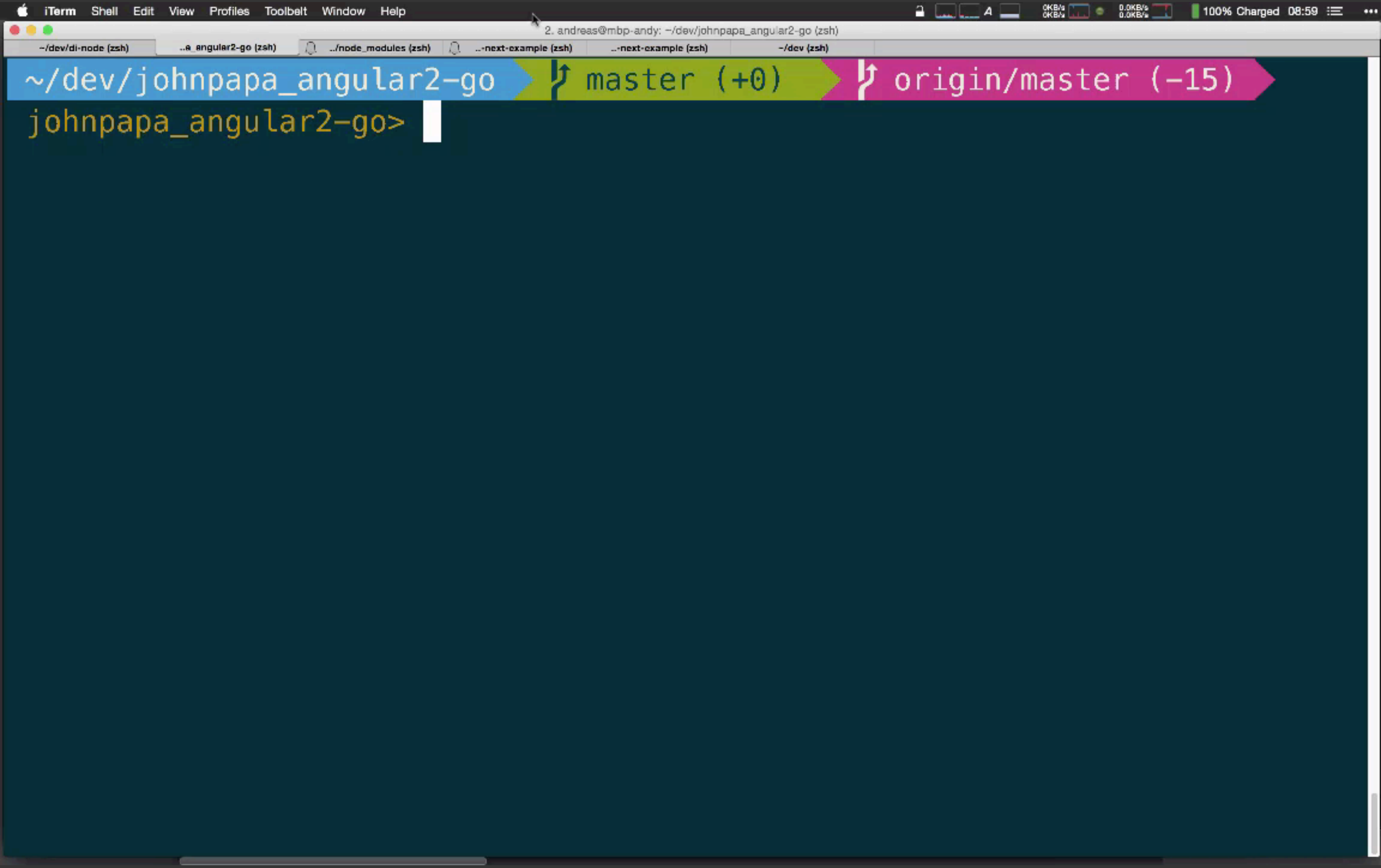
Refactoring (e.g. Renaming)

IDE Tooling



```
import {Person} from './character';

export class CharacterService {
  getCharacters() { return Promise.resolve(CHARACTERS); }

  getCharacter(id: number) {
    return Promise.resolve(CHARACTERS)
      .then((characters) => { return characters.filter((c) => {
        return c.id === id;
      })[0]});
  }
}

var CHARACTERS : Person[] = [
```

Refactoring (e.g. Renaming)

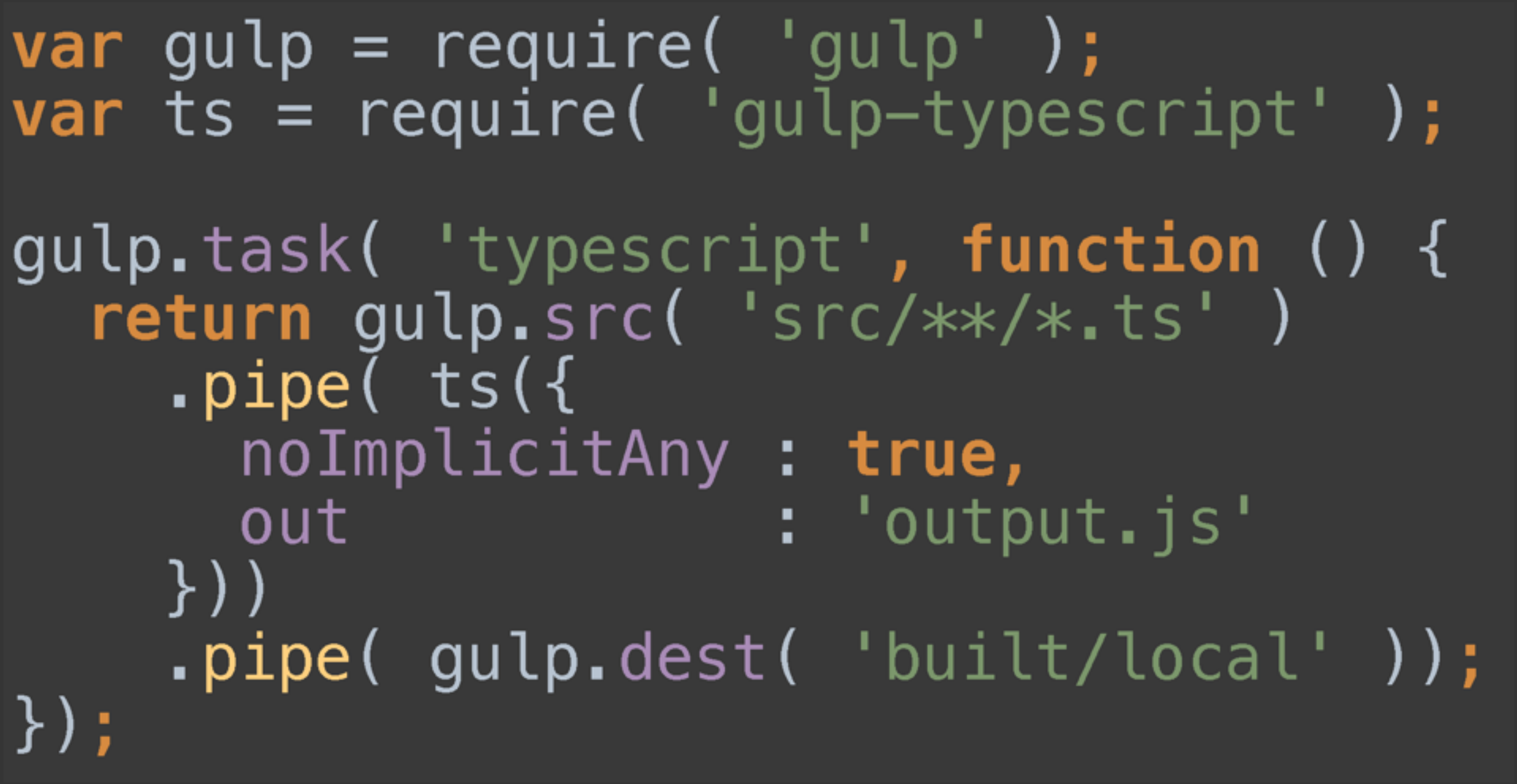


```
~/dev/johnpapa_angular2-go   master (+0)   origin/master (-15)
johnpapa_angular2-go>
```

Task Runner

Gulp

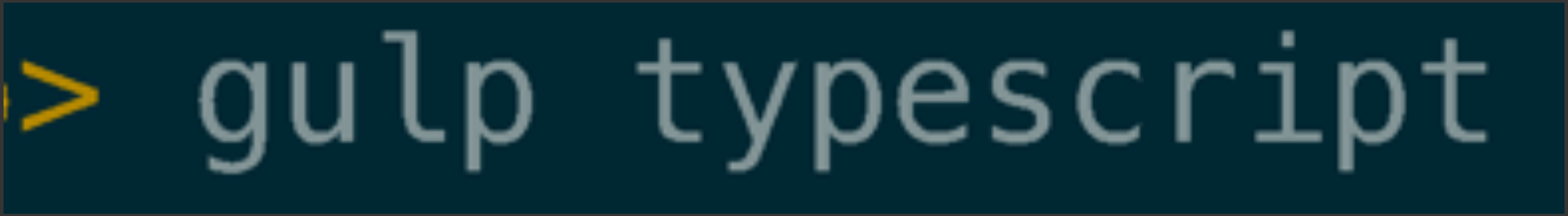
Gulp Task



```
var gulp = require( 'gulp' );
var ts = require( 'gulp-typescript' );

gulp.task( 'typescript', function () {
  return gulp.src( 'src/**/*.ts' )
    .pipe( ts({
      noImplicitAny : true,
      out           : 'output.js'
    }))
    .pipe( gulp.dest( 'built/local' ));
});
```

Compile TypeScript

Running a Gulp Task

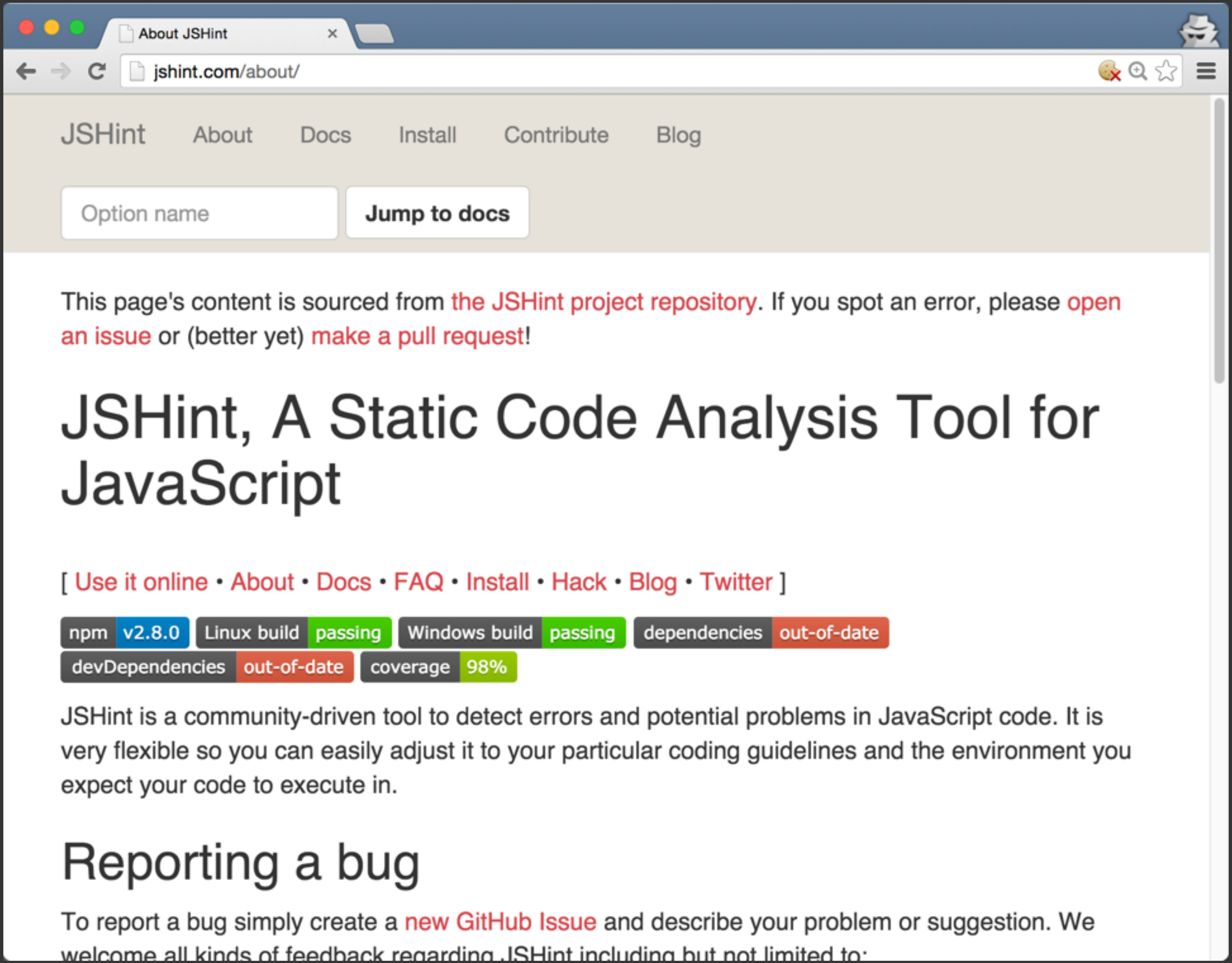


```
> gulp typescript
```

Static Code Analysis

JSHint, ESLint, TSLint

JSHint



This page's content is sourced from the JSHint project repository. If you spot an error, please open an issue or (better yet) make a pull request!

JSHint, A Static Code Analysis Tool for JavaScript

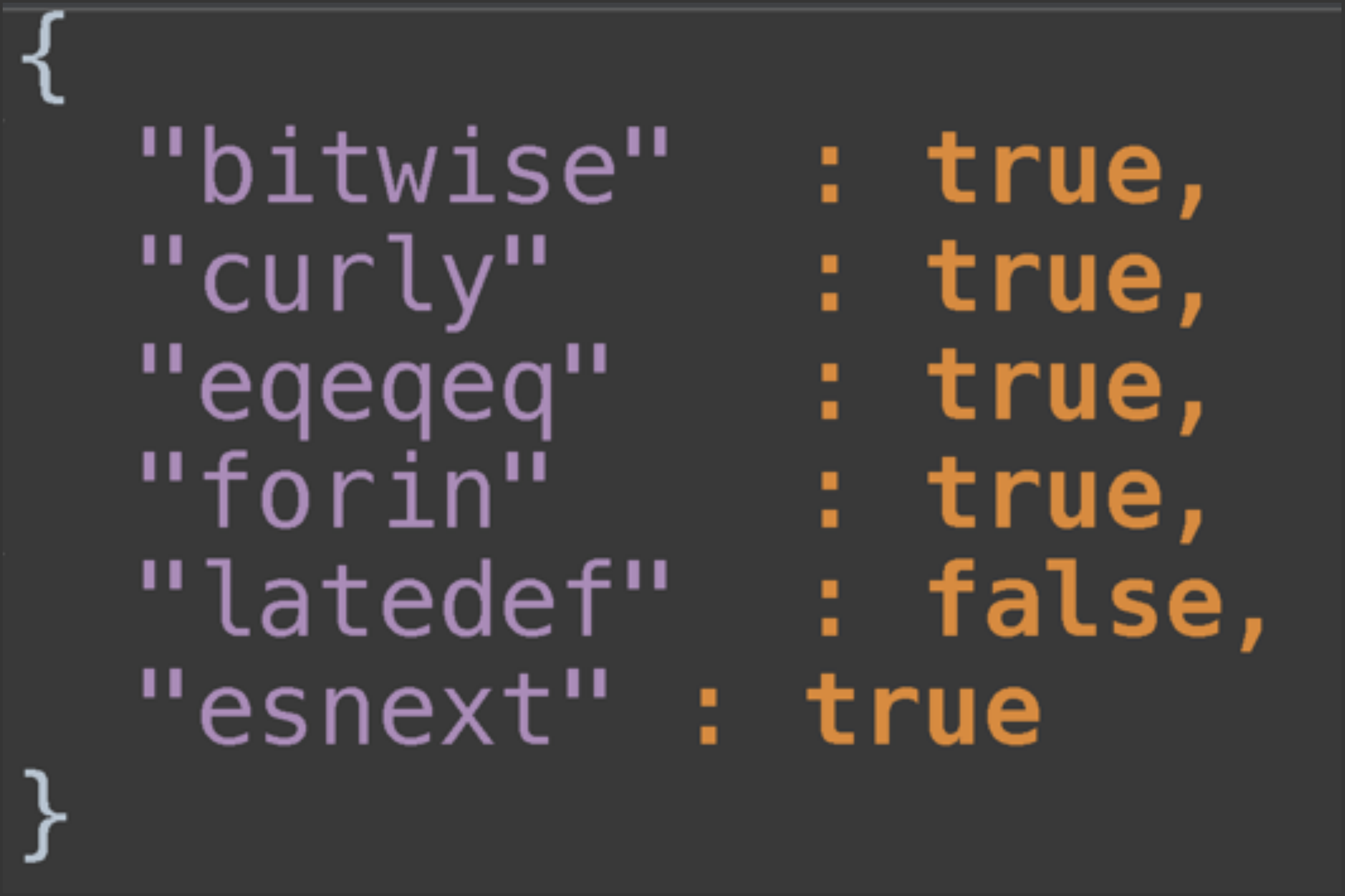
[Use it online • About • Docs • FAQ • Install • Hack • Blog • Twitter]

JSHint is a community-driven tool to detect errors and potential problems in JavaScript code. It is very flexible so you can easily adjust it to your particular coding guidelines and the environment you expect your code to execute in.

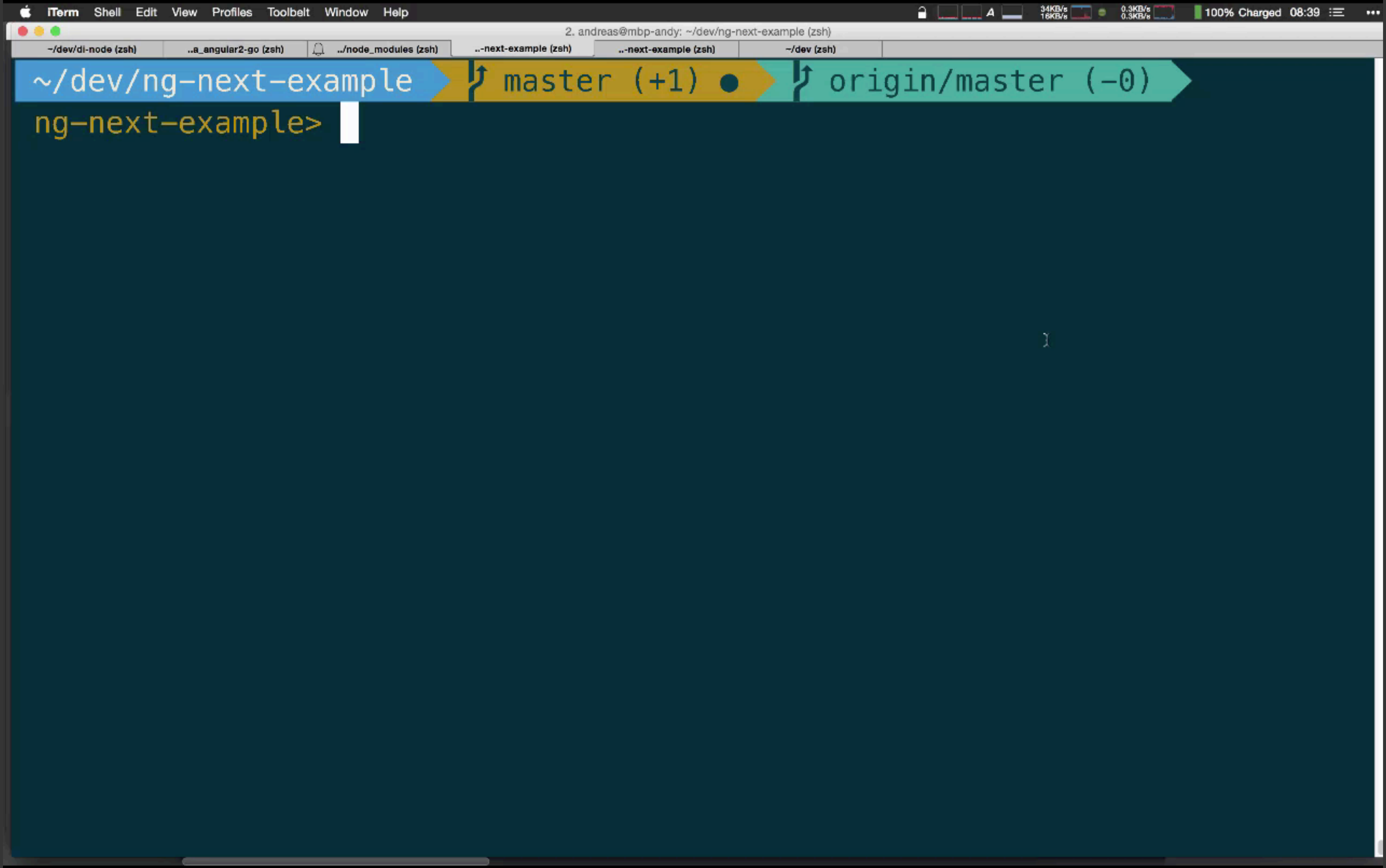
Reporting a bug

To report a bug simply create a new GitHub Issue and describe your problem or suggestion. We welcome all kinds of feedback regarding JSHint including but not limited to:

.jshintrc

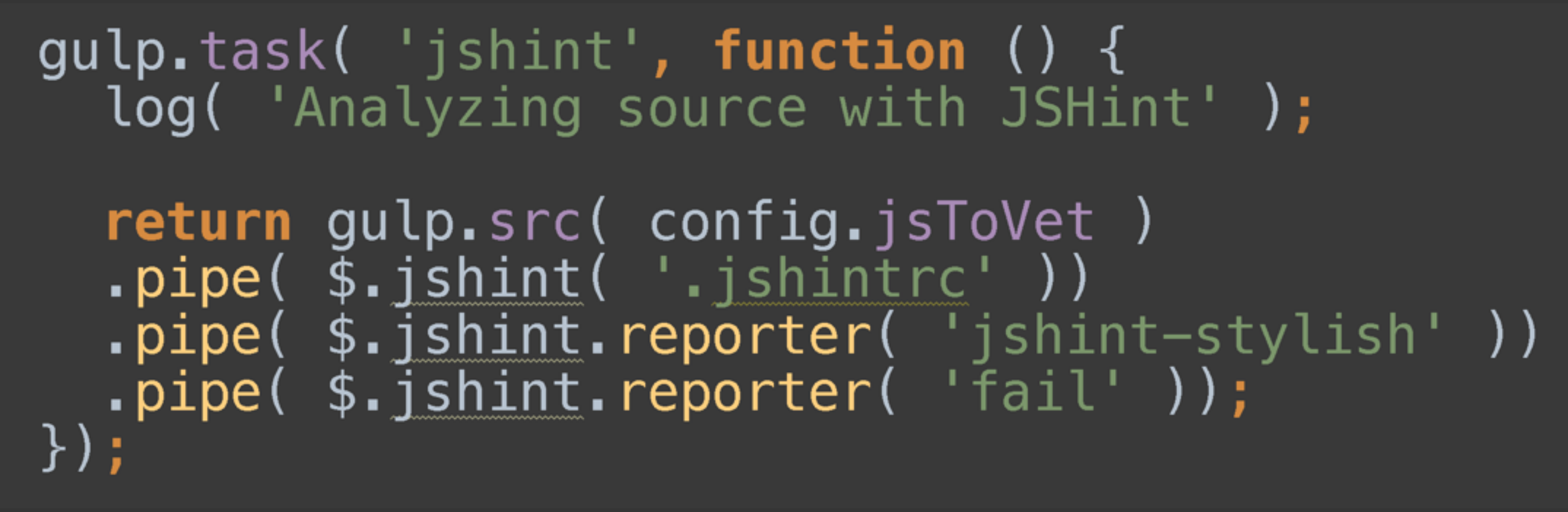


```
{
  "bitwise"  : true,
  "curly"    : true,
  "eqeqeq"   : true,
  "forin"    : true,
  "latedef"  : false,
  "esnext" : true
}
```



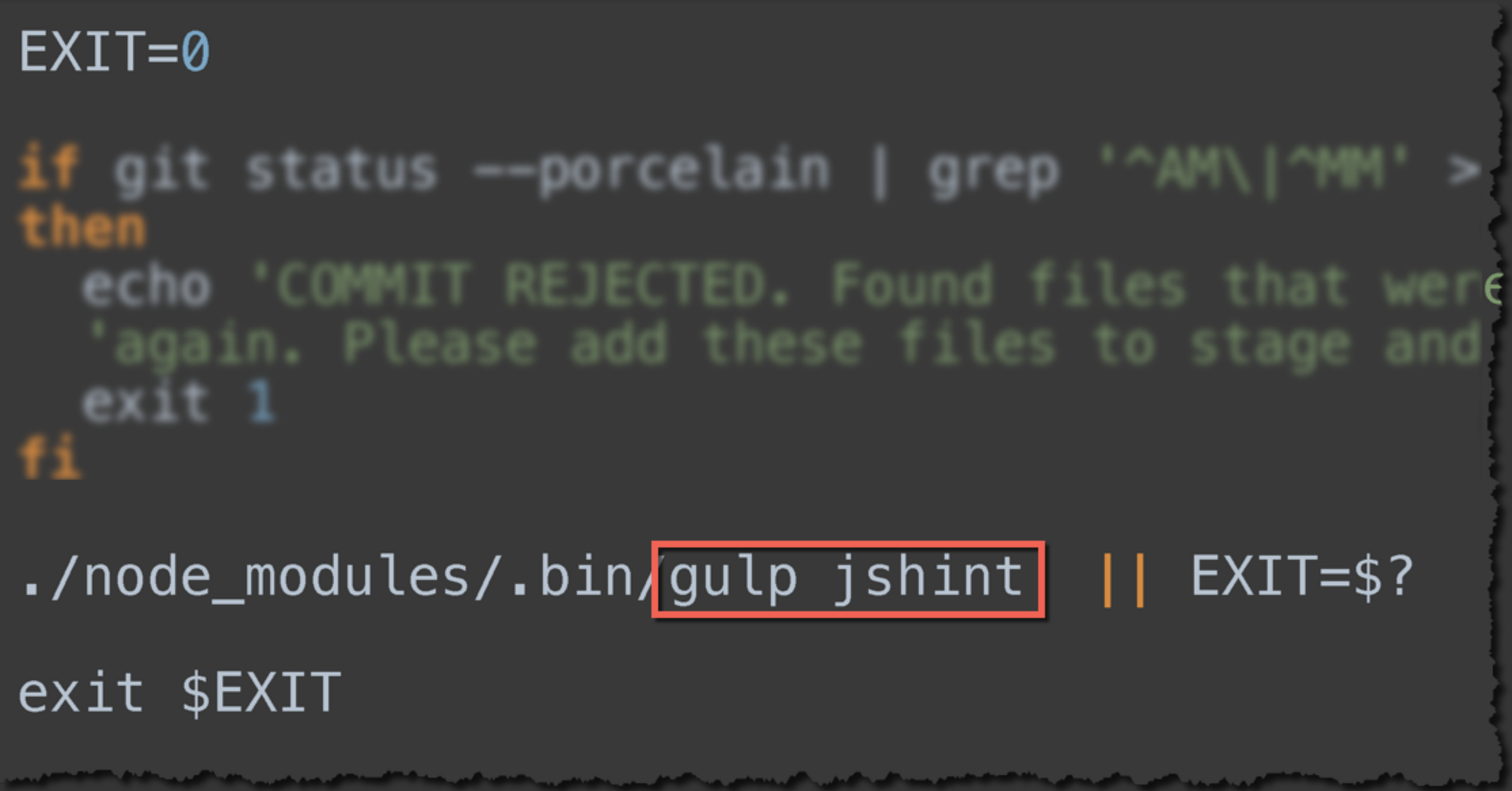
```
~/dev/ng-next-example  master (+1) ●  origin/master (-0)
ng-next-example>
```

JSHint Gulp Task



```
gulp.task( 'jshint', function () {
  log( 'Analyzing source with JSHint' );

  return gulp.src( config.jsToVet )
  .pipe( $.jshint( '.jshintrc' ))
  .pipe( $.jshint.reporter( 'jshint-stylish' ))
  .pipe( $.jshint.reporter( 'fail' ));
});
```

Pre Commit Hook



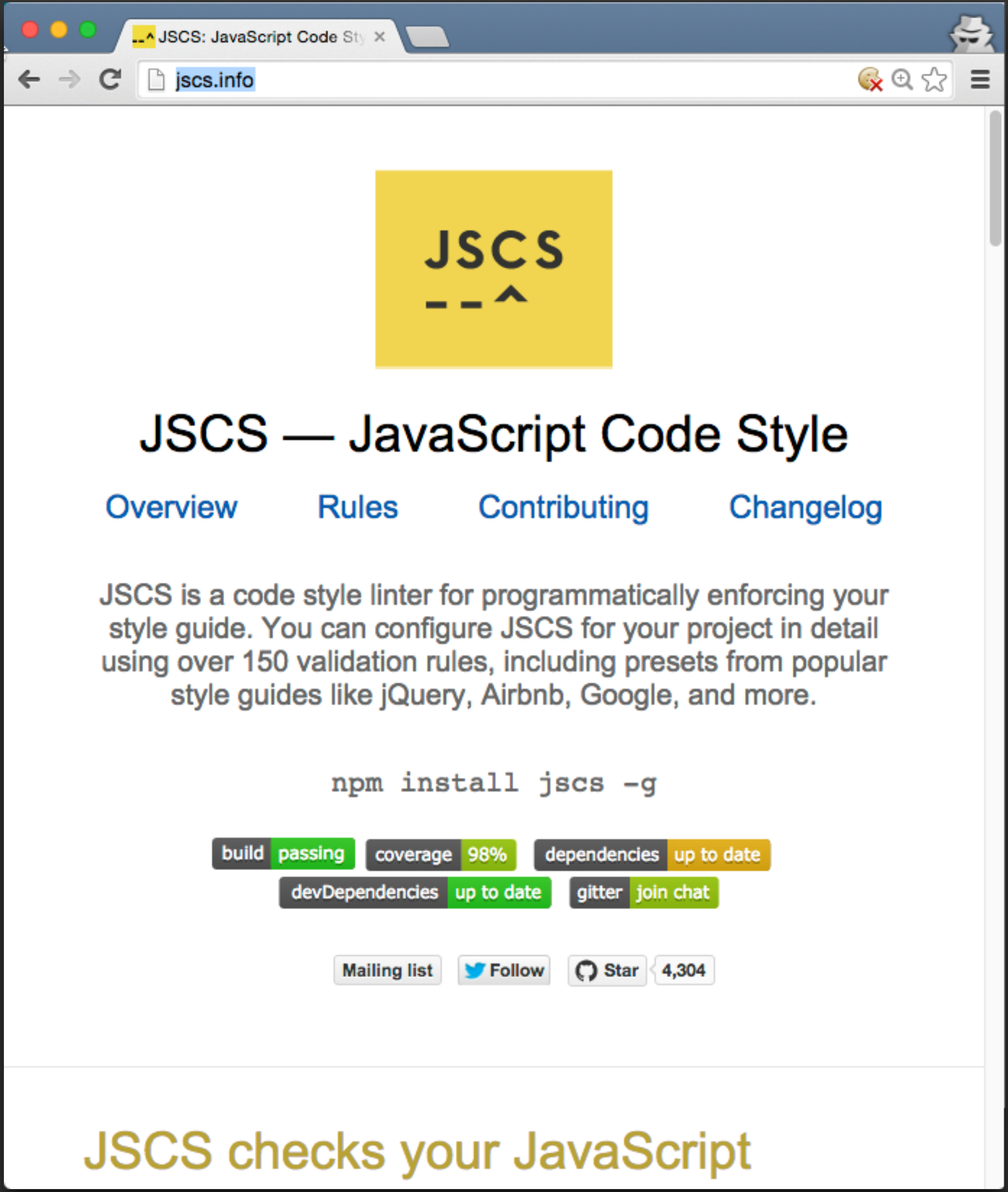
```
EXIT=0

if git status --porcelain | grep '^AM\|^MM' >
then
   echo 'COMMIT REJECTED. Found files that were
    'again. Please add these files to stage and
   exit 1
fi

./node_modules/.bin/gulp jshint  || EXIT=$?

exit $EXIT
```

Executes Specific (Gulp) Tasks on a Commit

Naming

Functions

Form / Style



JSCS — JavaScript Code Style

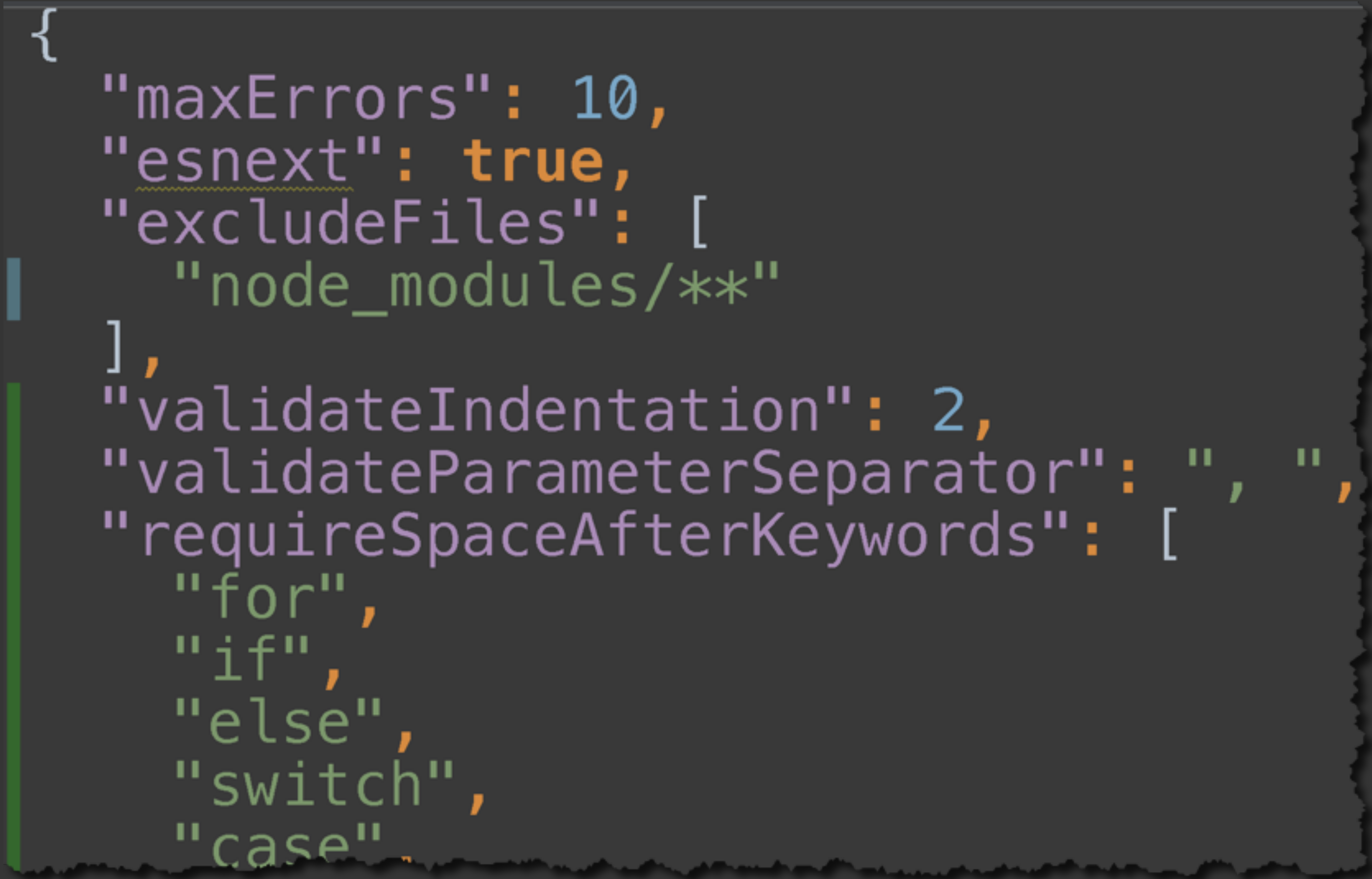
JSCS is a code style linter for programmatically enforcing your style guide. You can configure JSCS for your project in detail using over 150 validation rules, including presets from popular style guides like jQuery, Airbnb, Google, and more.

```
npm install jscs -g
```

JSCS checks your JavaScript

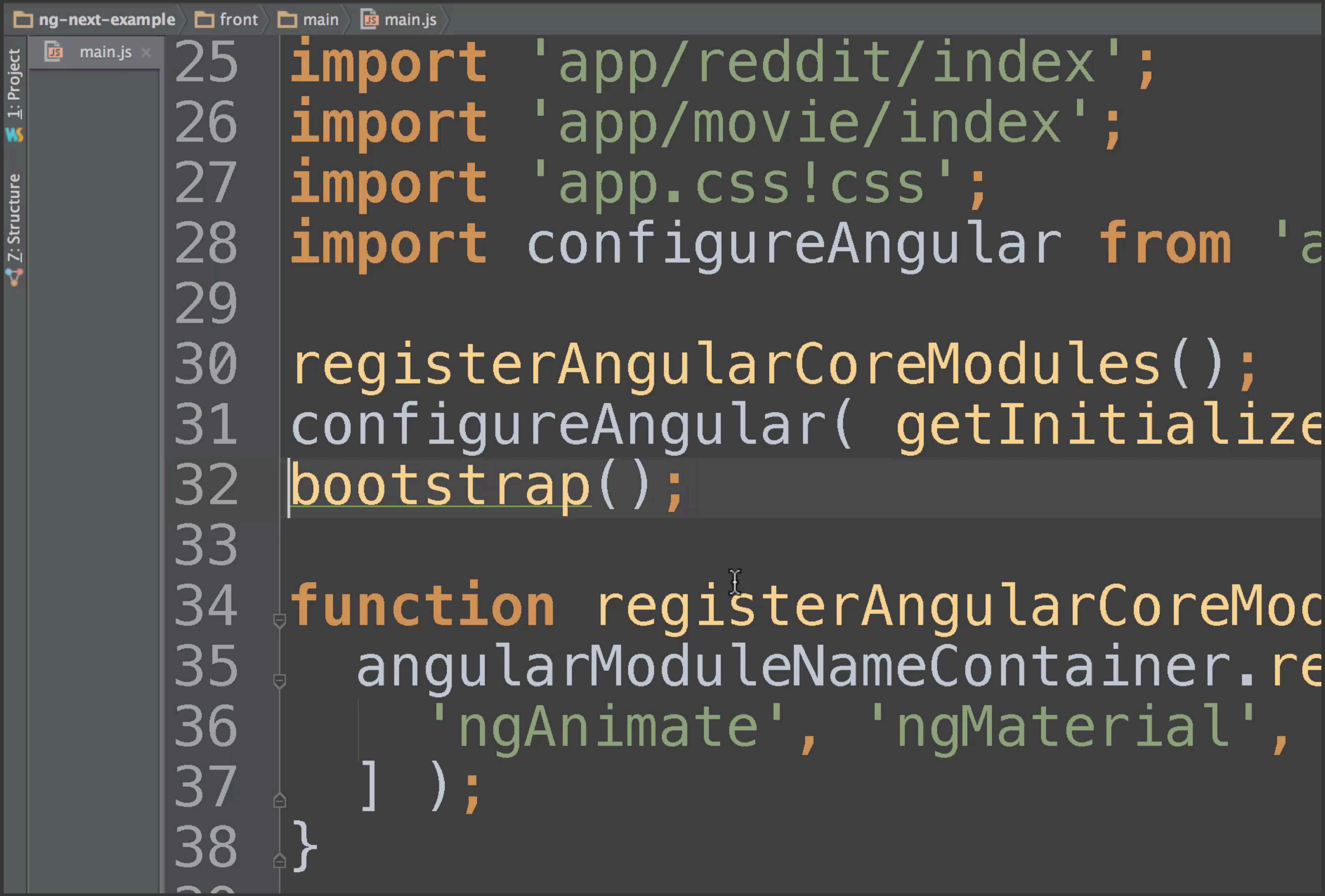
http://jscs.info/

.jscsrc



```
{
  "maxErrors": 10,
  "esnext": true,
  "excludeFiles": [
    "node_modules/**"
  ],
  "validateIndentation": 2,
  "validateParameterSeparator": ", ",
  "requireSpaceAfterKeywords": [
    "for",
    "if",
    "else",
    "switch",
    "case"
```

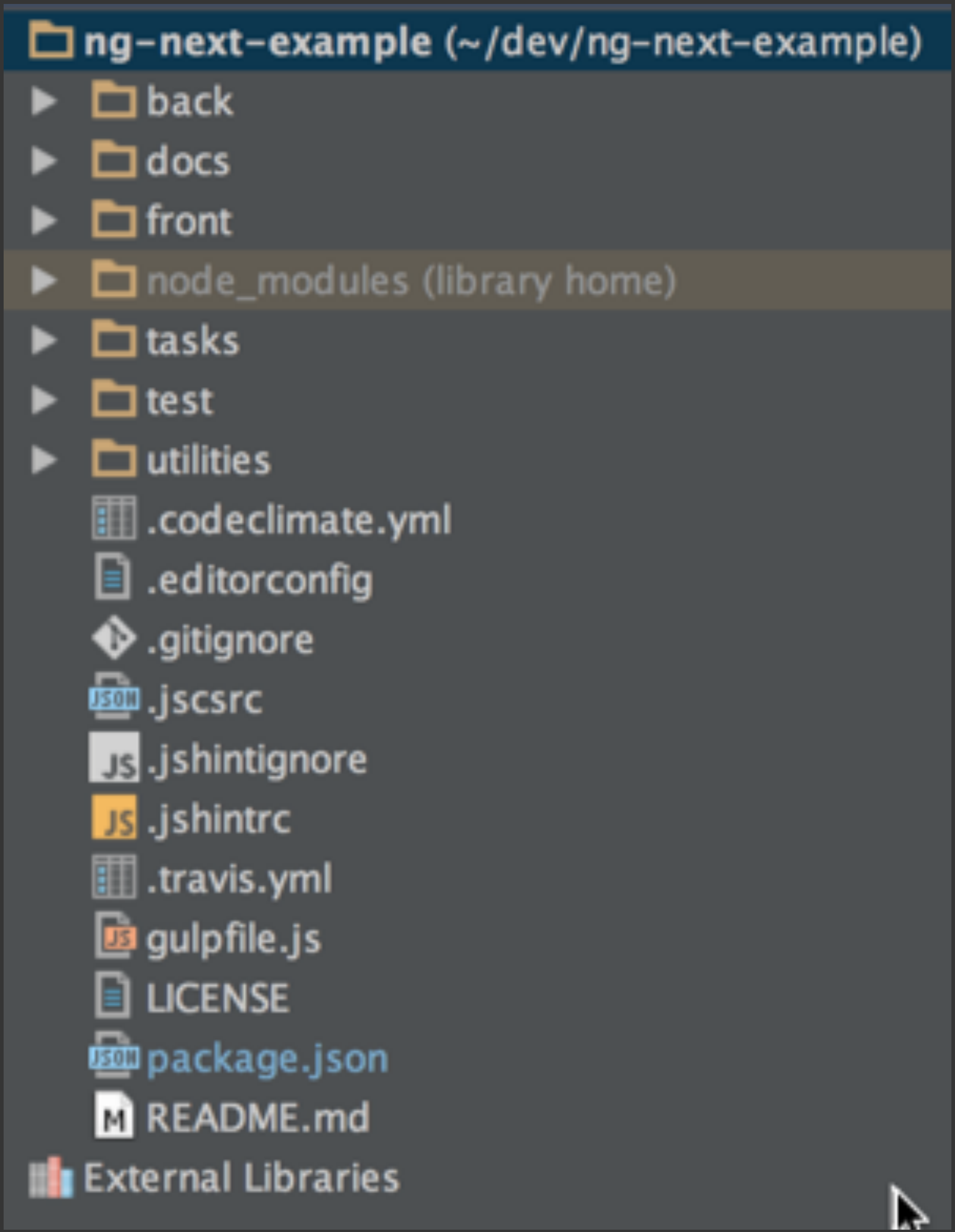
JSCS IDE Support



```
import 'app/reddit/index';
import 'app/movie/index';
import 'app.css!css';
import configureAngular from 'a

registerAngularCoreModules();
configureAngular( getInitialize
bootstrap();

function registerAngularCoreMod
  angularModuleNameContainer.re
    'ngAnimate', 'ngMaterial',
  ] );
}
```

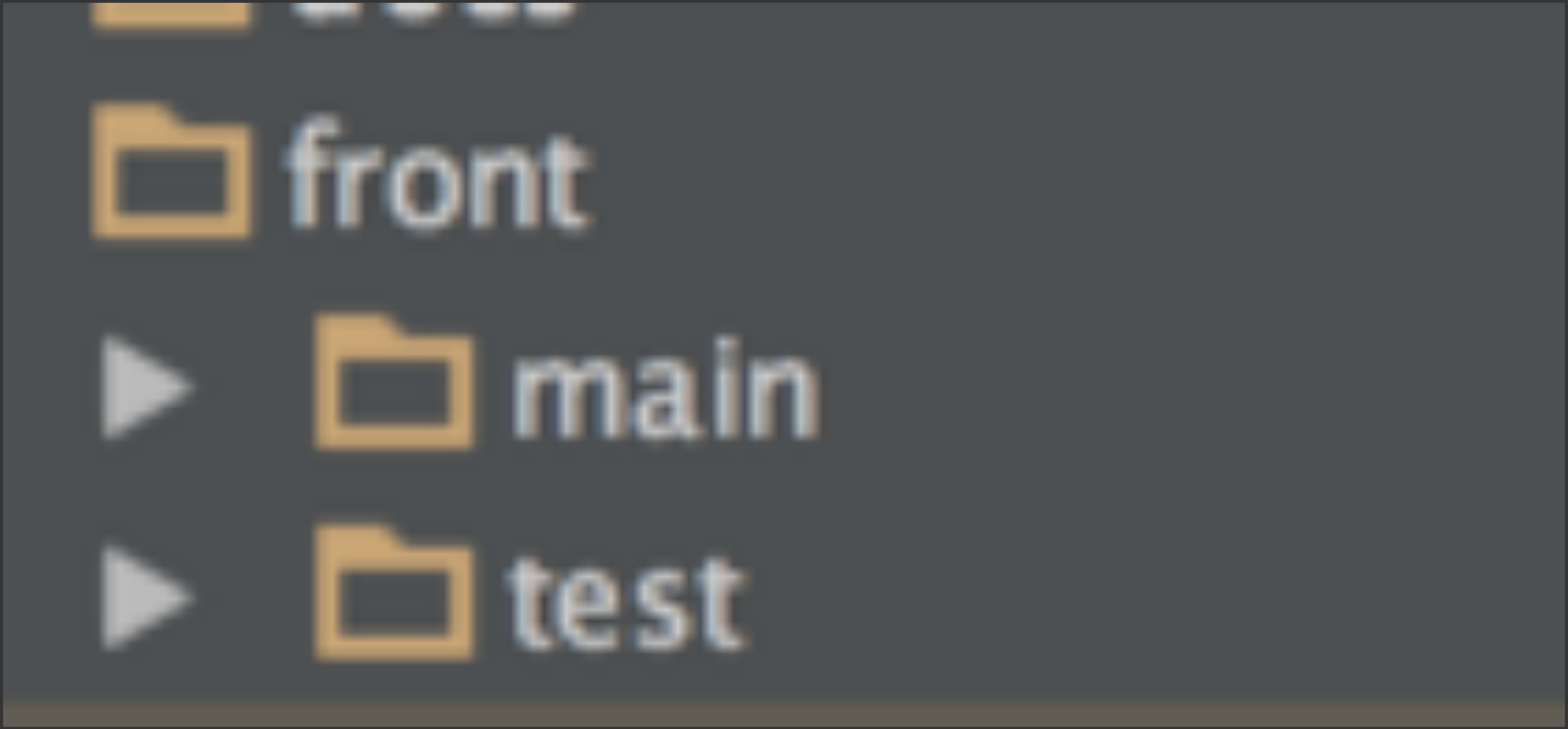
Folder Structure

Should Reveal the Domain

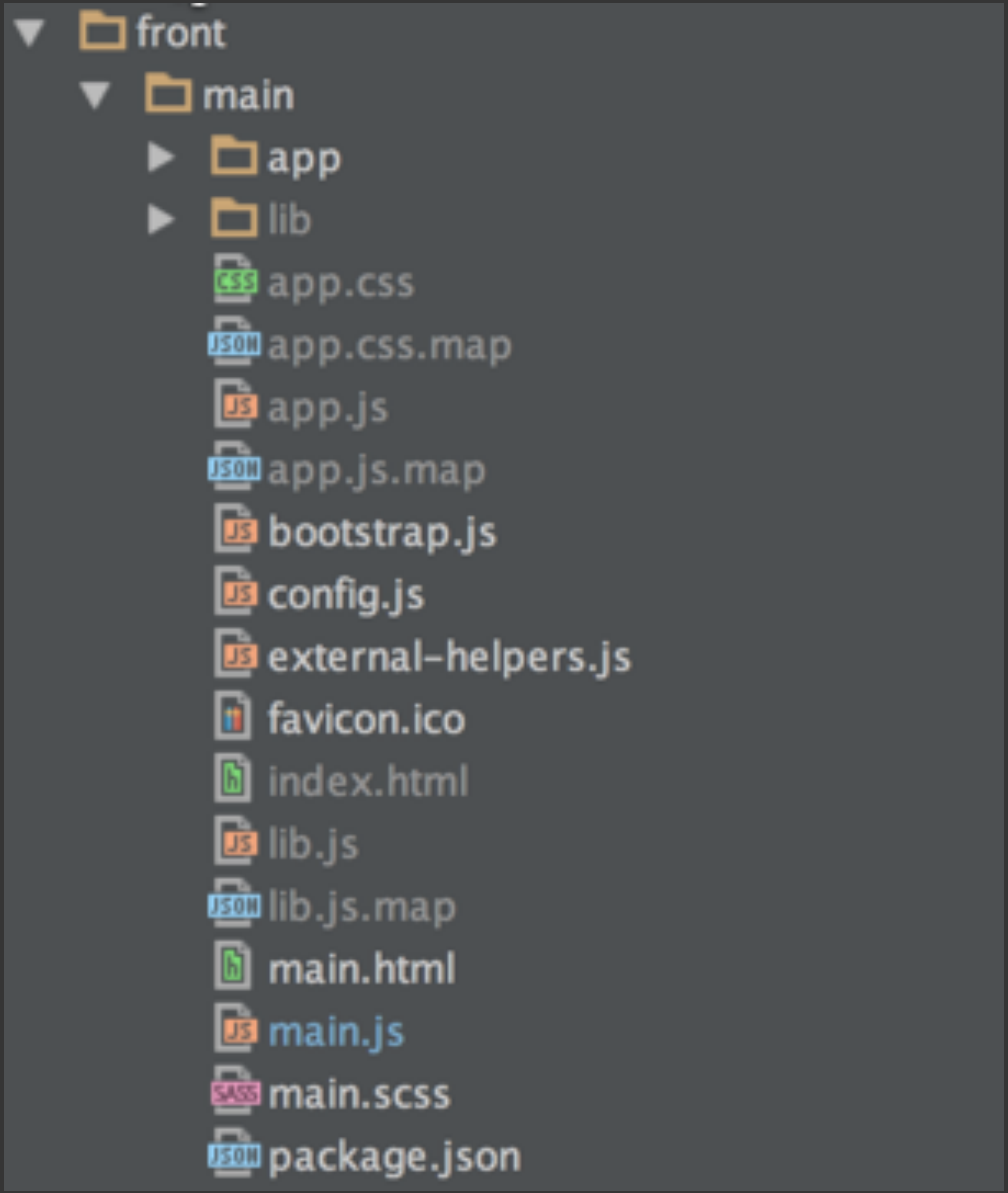


```
ng-next-example (~/dev/ng-next-example)
  ▶ back
  ▶ docs
  ▶ front
  ▶ node_modules (library home)
  ▶ tasks
  ▶ test
  ▶ utilities
    .codeclimate.yml
    .editorconfig
    .gitignore
    .jscsrc
    .jshintignore
    .jshintrc
    .travis.yml
    gulpfile.js
    LICENSE
    package.json
    README.md
External Libraries
```

Repository Root

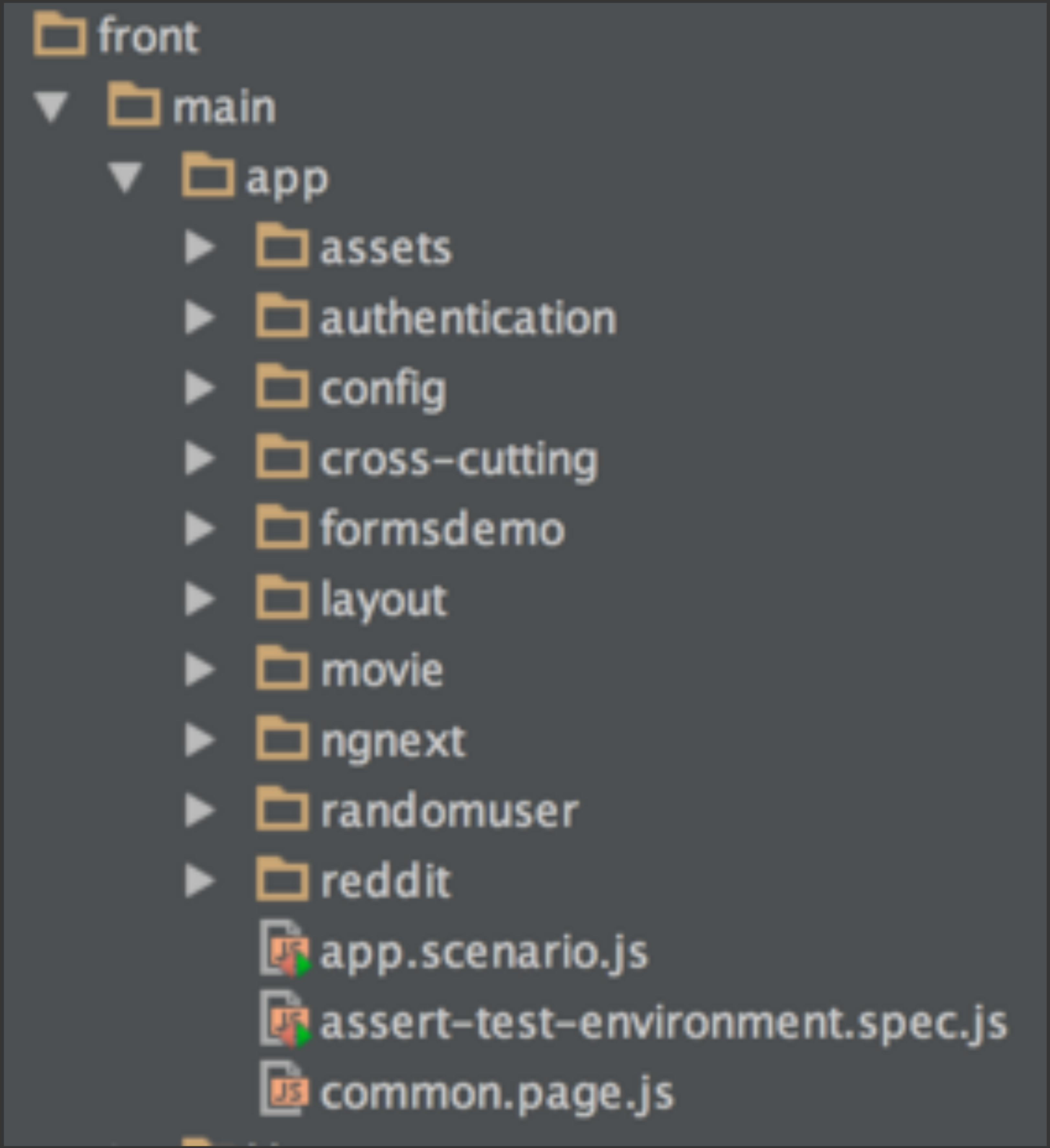


Front-End



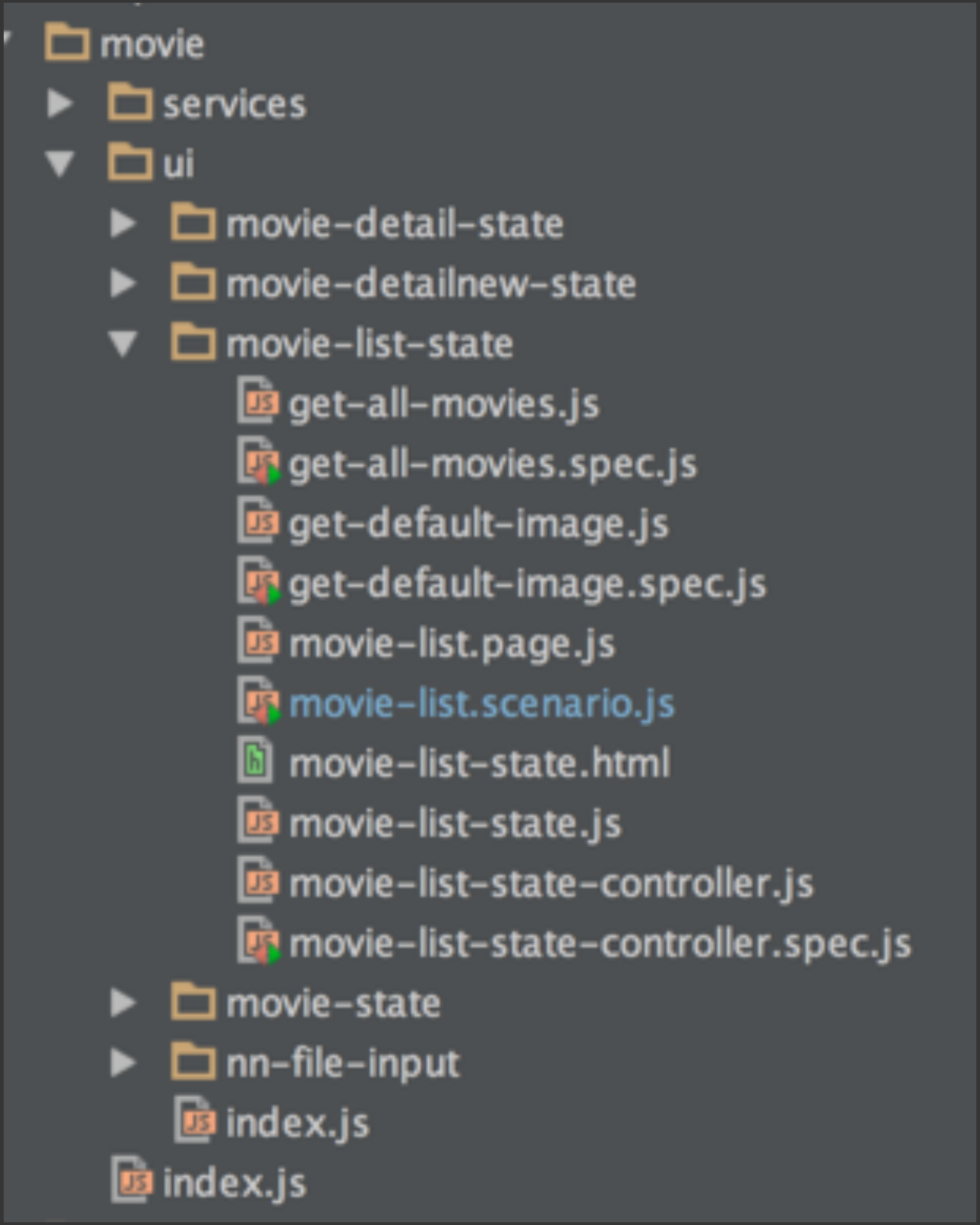
```
▼ front
  ▼ main
    ▶ app
    ▶ lib
      app.css
      app.css.map
      app.js
      app.js.map
      bootstrap.js
      config.js
      external-helpers.js
      favicon.ico
      index.html
      lib.js
      lib.js.map
      main.html
      main.js
      main.scss
      package.json
```

Main Partition (most concrete)



```
front
▼ main
  ▼ app
    ▶ assets
    ▶ authentication
    ▶ config
    ▶ cross-cutting
    ▶ formsdemo
    ▶ layout
    ▶ movie
    ▶ ngnext
    ▶ randomuser
    ▶ reddit
      app.scenario.js
      assert-test-environment.spec.js
      common.page.js
```

Components



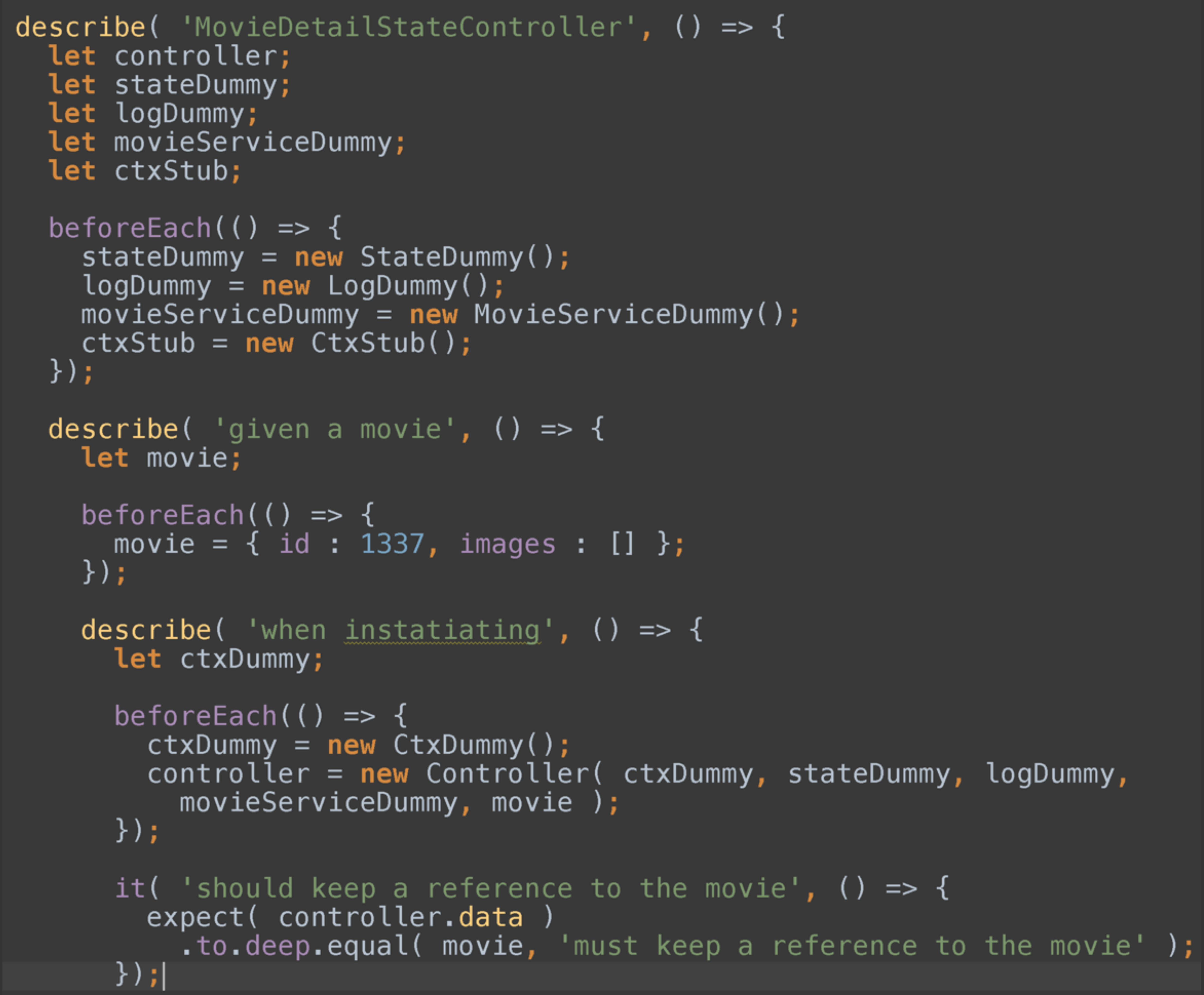
```
▼ movie
    ▶ services
    ▼ ui
        ▶ movie-detail-state
        ▶ movie-detailnew-state
        ▼ movie-list-state
              get-all-movies.js
              get-all-movies.spec.js
              get-default-image.js
              get-default-image.spec.js
              movie-list.page.js
              movie-list.scenario.js
              movie-list-state.html
              movie-list-state.js
              movie-list-state-controller.js
              movie-list-state-controller.spec.js
        ▶ movie-state
        ▶ nn-file-input
          index.js
      index.js
```

Main Partition (most concrete)

Testing

Unit Testing

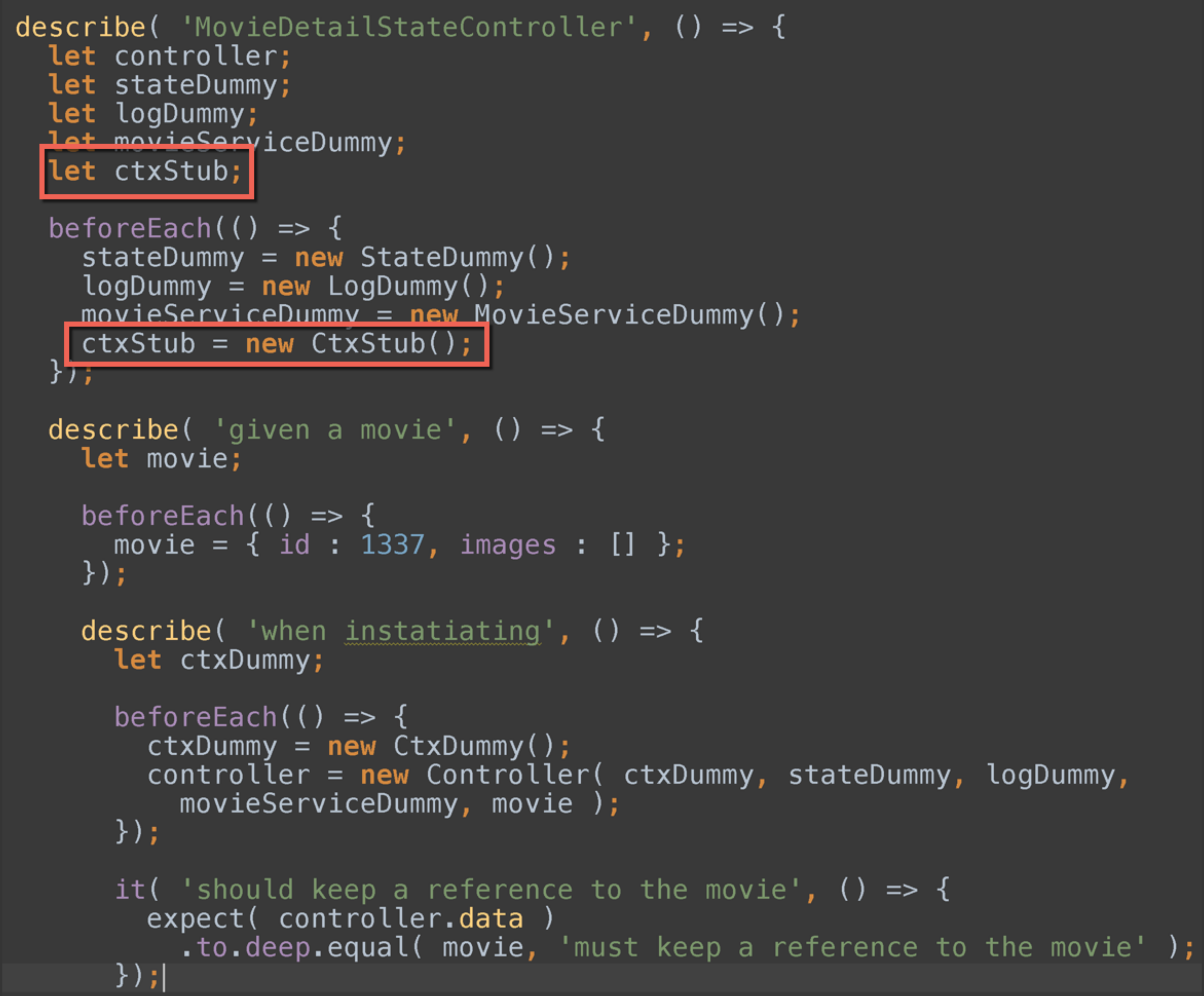
Jasmine, Mocha



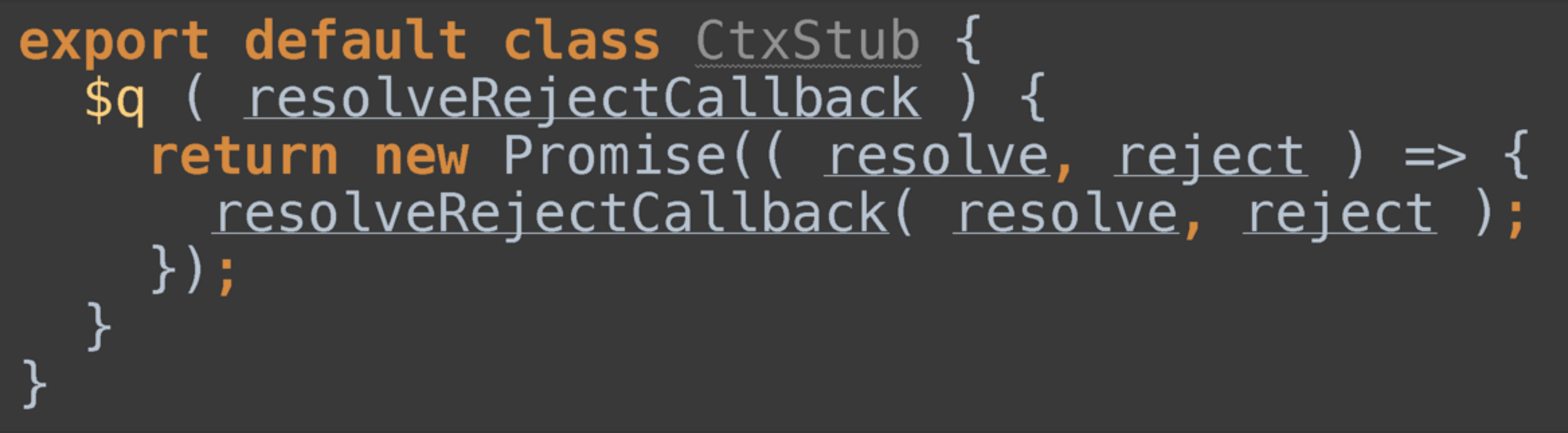
```
describe( 'MovieDetailStateController', () => {
  let controller;
  let stateDummy;
  let logDummy;
  let movieServiceDummy;
  let ctxStub;

  beforeEach(() => {
    stateDummy = new StateDummy();
    logDummy = new LogDummy();
    movieServiceDummy = new MovieServiceDummy();
    ctxStub = new CtxStub();
  });

  describe( 'given a movie', () => {
    let movie;

    beforeEach(() => {
      movie = { id : 1337, images : [] };
    });

    describe( 'when instatiating', () => {
      let ctxDummy;

      beforeEach(() => {
        ctxDummy = new CtxDummy();
        controller = new Controller( ctxDummy, stateDummy, logDummy,
          movieServiceDummy, movie );
      });

      it( 'should keep a reference to the movie', () => {
        expect( controller.data )
          .to.deep.equal( movie, 'must keep a reference to the movie' );
      });
```

Mocking Libraries

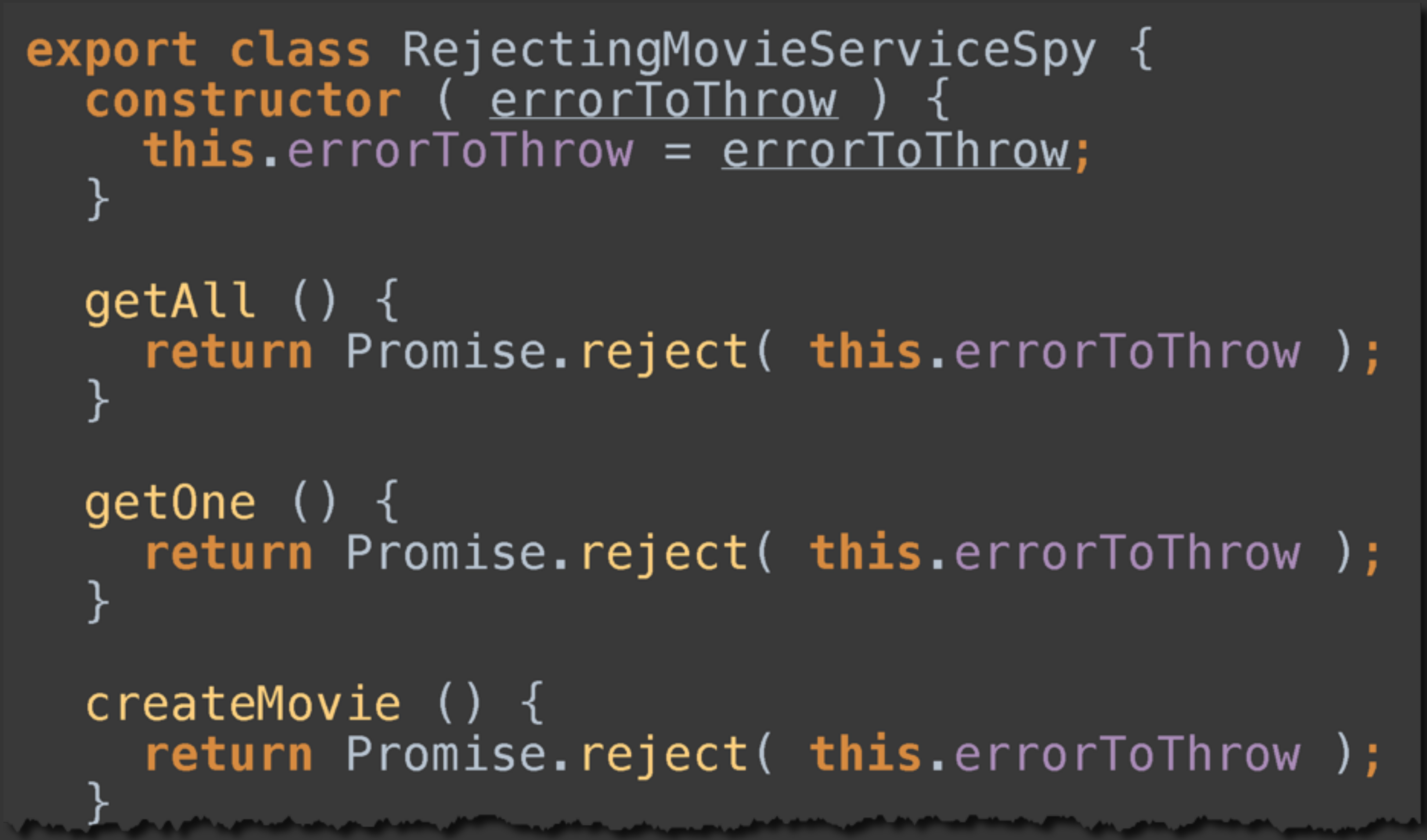
Sinon.js, DYI



```
describe( 'MovieDetailStateController', () => {
  let controller;
  let stateDummy;
  let logDummy;
  let movieServiceDummy;
  let ctxStub;

  beforeEach(() => {
    stateDummy = new StateDummy();
    logDummy = new LogDummy();
    movieServiceDummy = new MovieServiceDummy();
    ctxStub = new CtxStub();
  });

  describe( 'given a movie', () => {
    let movie;

    beforeEach(() => {
      movie = { id : 1337, images : [] };
    });

    describe( 'when instatiating', () => {
      let ctxDummy;

      beforeEach(() => {
        ctxDummy = new CtxDummy();
        controller = new Controller( ctxDummy, stateDummy, logDummy,
          movieServiceDummy, movie );
      });

      it( 'should keep a reference to the movie', () => {
        expect( controller.data )
          .to.deep.equal( movie, 'must keep a reference to the movie' );
      });
```



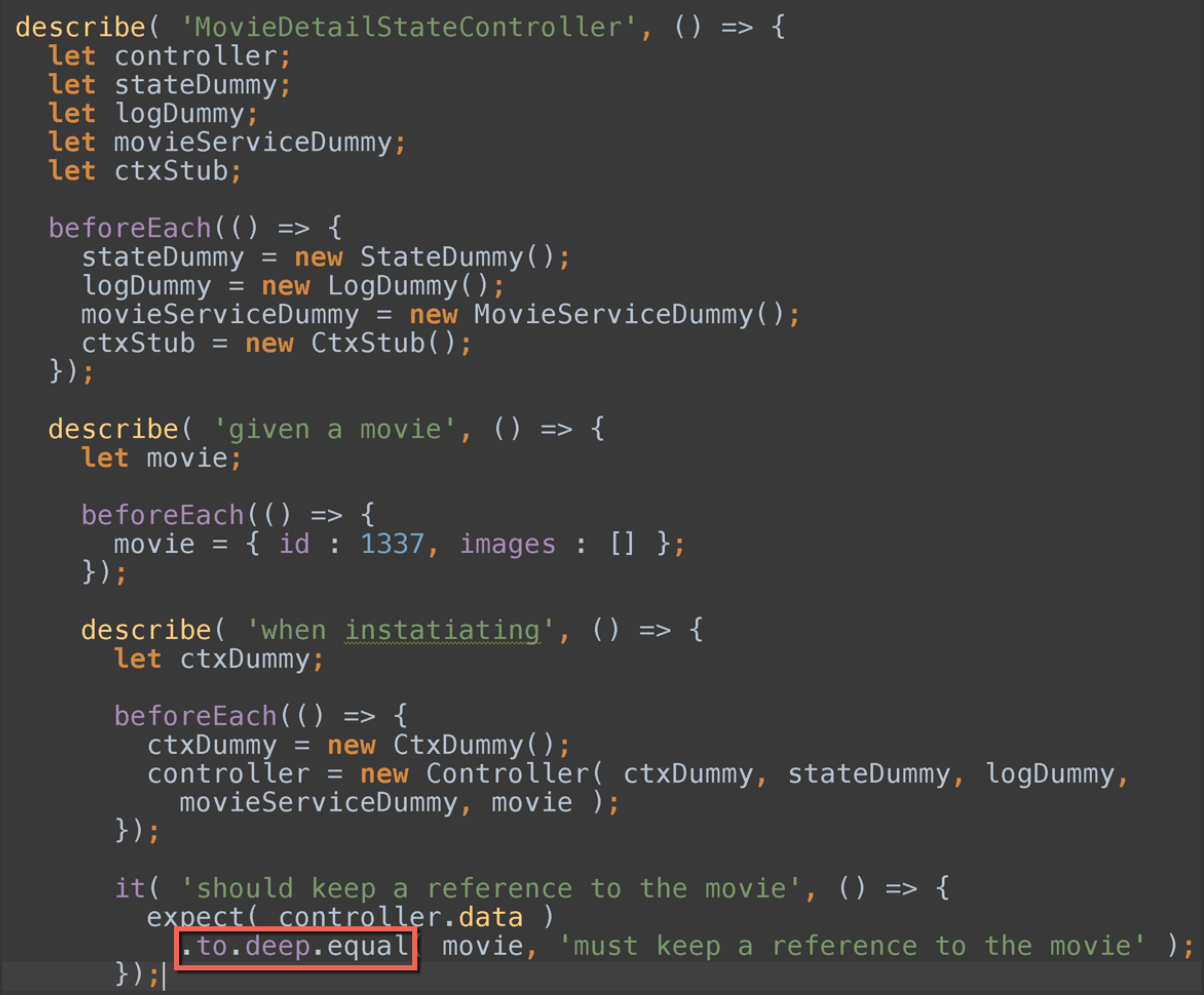
```
export default class CtxStub {
  $q ( resolveRejectCallback ) {
    return new Promise(( resolve, reject ) => {
      resolveRejectCallback( resolve, reject );
    });
  }
}
```



```
export class RejectingMovieServiceSpy {
  constructor ( errorToThrow ) {
    this.errorToThrow = errorToThrow;
  }

  getAll () {
    return Promise.reject( this.errorToThrow );
  }

  getOne () {
    return Promise.reject( this.errorToThrow );
  }

  createMovie () {
    return Promise.reject( this.errorToThrow );
  }
```

Assertion Libraries

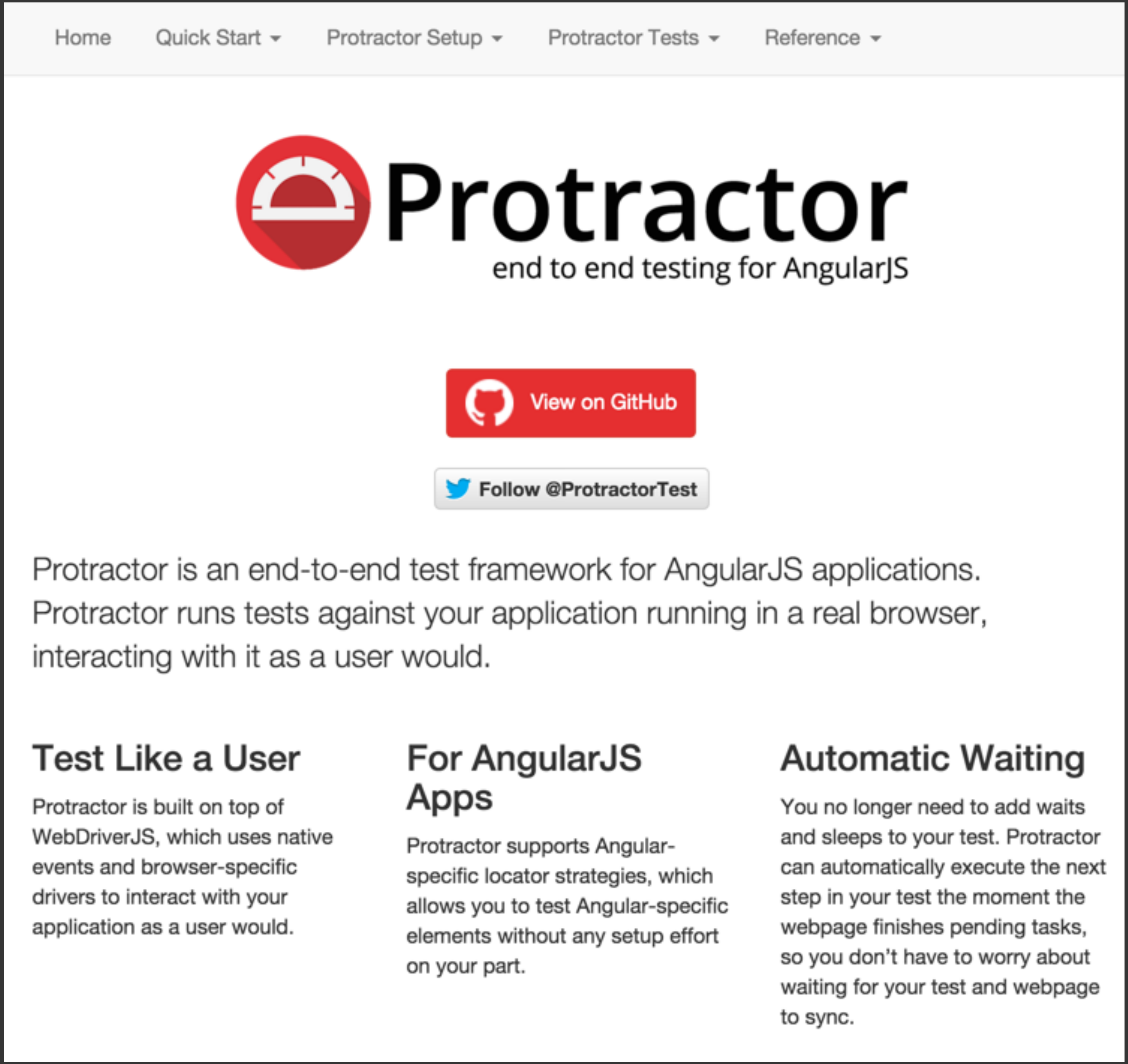
Chai



```
describe( 'MovieDetailStateController', () => {
  let controller;
  let stateDummy;
  let logDummy;
  let movieServiceDummy;
  let ctxStub;

  beforeEach(() => {
    stateDummy = new StateDummy();
    logDummy = new LogDummy();
    movieServiceDummy = new MovieServiceDummy();
    ctxStub = new CtxStub();
  });

  describe( 'given a movie', () => {
    let movie;

    beforeEach(() => {
      movie = { id : 1337, images : [] };
    });

    describe( 'when instatiating', () => {
      let ctxDummy;

      beforeEach(() => {
        ctxDummy = new CtxDummy();
        controller = new Controller( ctxDummy, stateDummy, logDummy,
          movieServiceDummy, movie );
      });

      it( 'should keep a reference to the movie', () => {
        expect( controller.data )
          .to.deep.equal( movie, 'must keep a reference to the movie' );
      });
```

End-To-End Testing

Protractor



Protractor

end to end testing for AngularJS

Protractor is an end-to-end test framework for AngularJS applications. Protractor runs tests against your application running in a real browser, interacting with it as a user would.

Test Like a User

Protractor is built on top of WebDriverJS, which uses native events and browser-specific drivers to interact with your application as a user would.

For AngularJS Apps

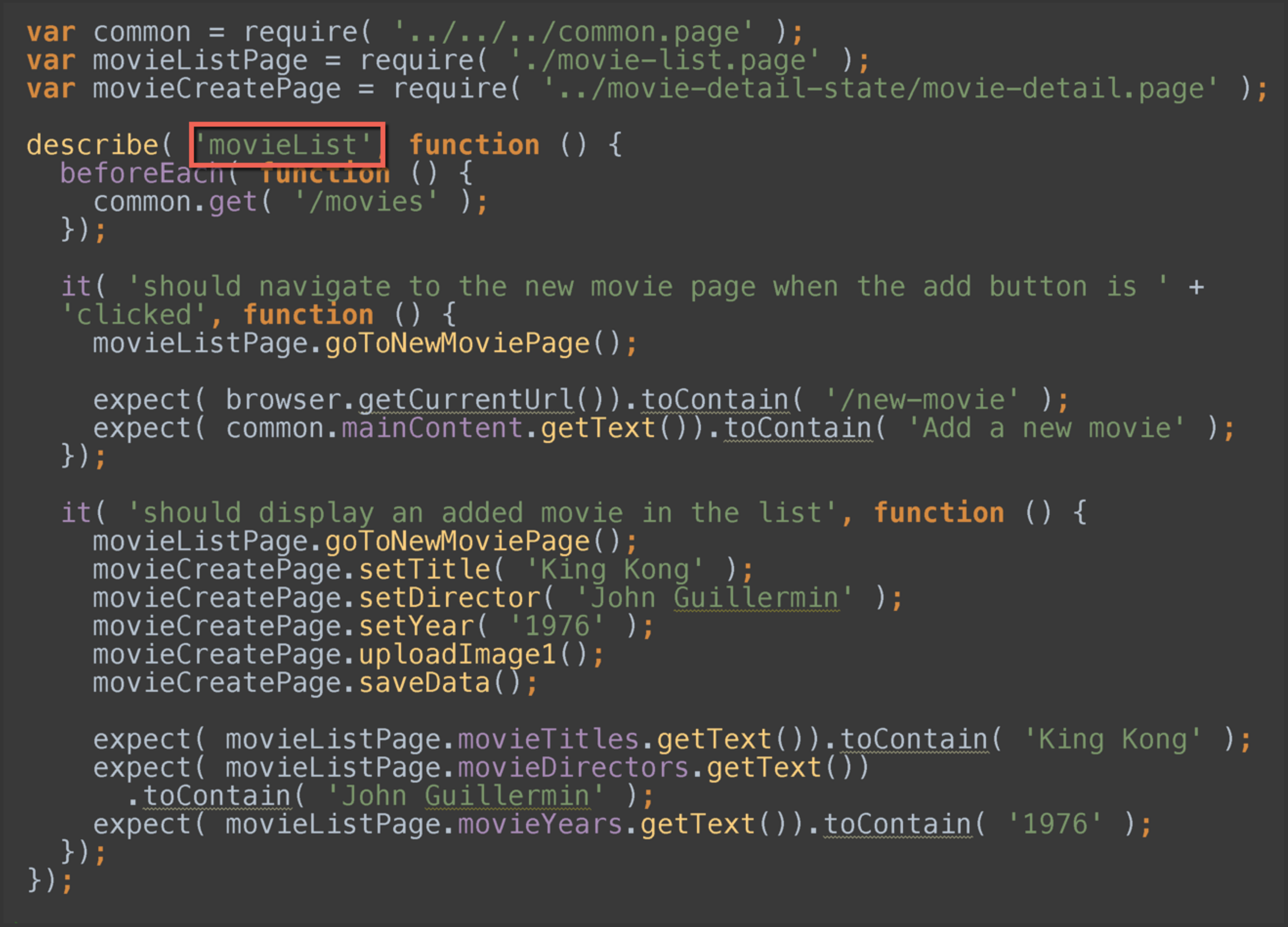
Protractor supports Angular-specific locator strategies, which allows you to test Angular-specific elements without any setup effort on your part.

Automatic Waiting

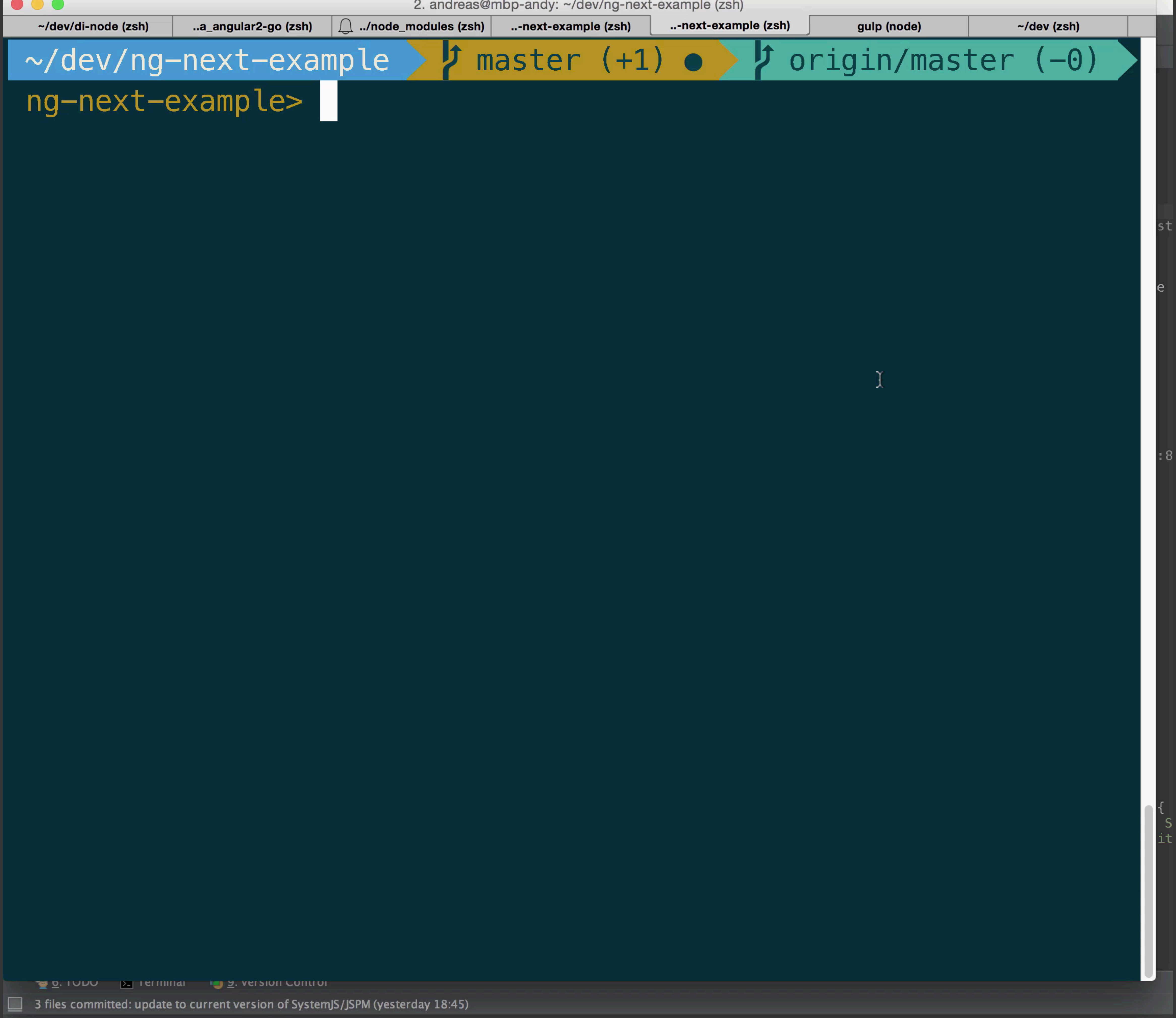
You no longer need to add waits and sleeps to your test. Protractor can automatically execute the next step in your test the moment the webpage finishes pending tasks, so you don't have to worry about waiting for your test and webpage to sync.

https://angular.github.io/protractor

Protractor



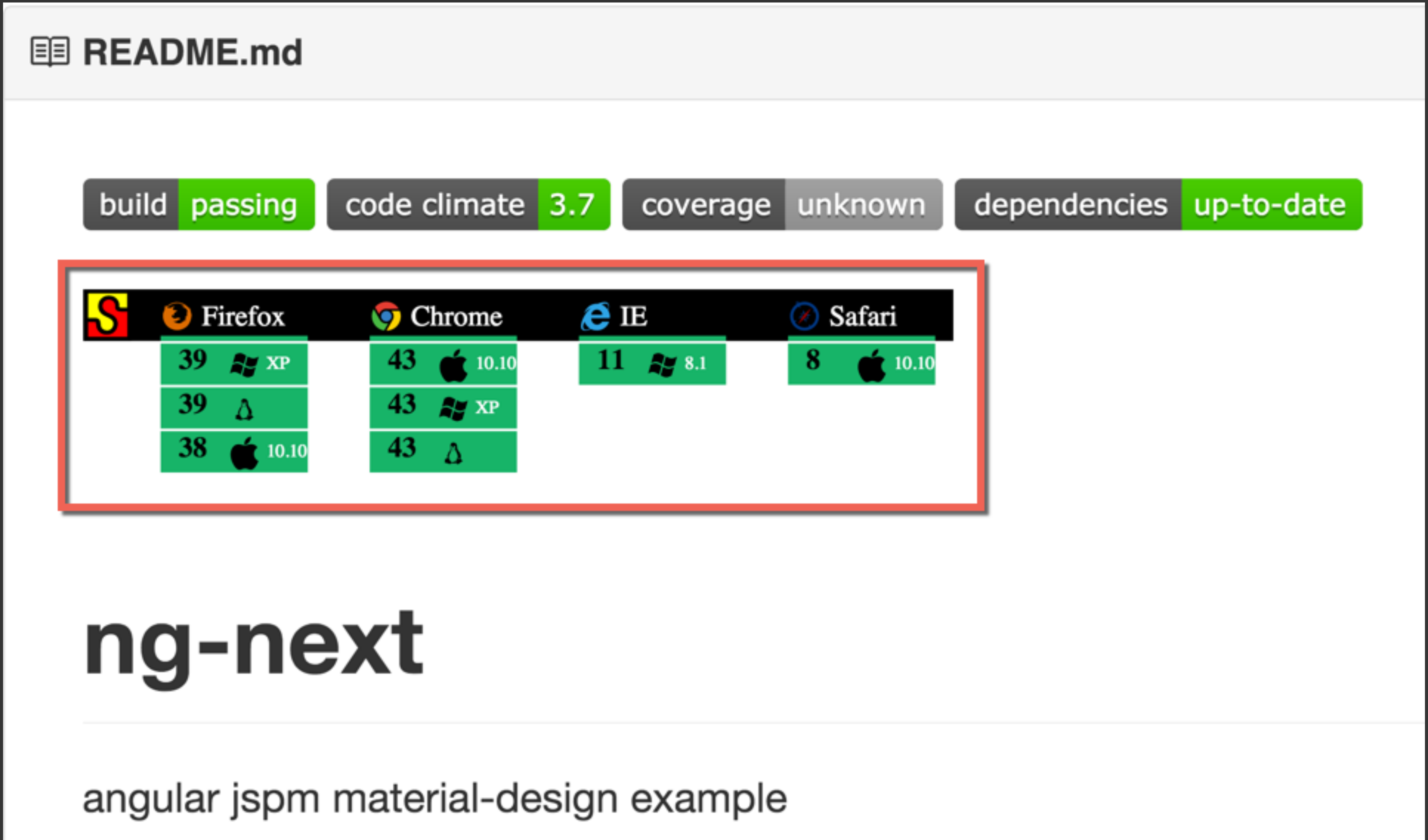
```
var common = require( '../../../common.page' );
var movieListPage = require( './movie-list.page' );
var movieCreatePage = require( '../movie-detail-state/movie-detail.page' );

describe( 'movieList', function () {
  beforeEach( function () {
    common.get( '/movies' );
  });

  it( 'should navigate to the new movie page when the add button is ' +
  'clicked', function () {
    movieListPage.goToNewMoviePage();

    expect( browser.getCurrentUrl()).toContain( '/new-movie' );
    expect( common.mainContent.getText()).toContain( 'Add a new movie' );
  });

  it( 'should display an added movie in the list', function () {
    movieListPage.goToNewMoviePage();
    movieCreatePage.setTitle( 'King Kong' );
    movieCreatePage.setDirector( 'John Guillermin' );
    movieCreatePage.setYear( '1976' );
    movieCreatePage.uploadImage1();
    movieCreatePage.saveData();

    expect( movieListPage.movieTitles.getText()).toContain( 'King Kong' );
    expect( movieListPage.movieDirectors.getText())
      .toContain( 'John Guillermin' );
    expect( movieListPage.movieYears.getText()).toContain( '1976' );
  });
});
```



```
~/dev/ng-next-example  master (+1) ●  origin/master (-0)
ng-next-example> 
```

3 files committed: update to current version of SystemJS/JSPM (yesterday 18:45)

E2E-Tests in the Cloud



README.md

build passing | code climate 3.7 | coverage unknown | dependencies up-to-date

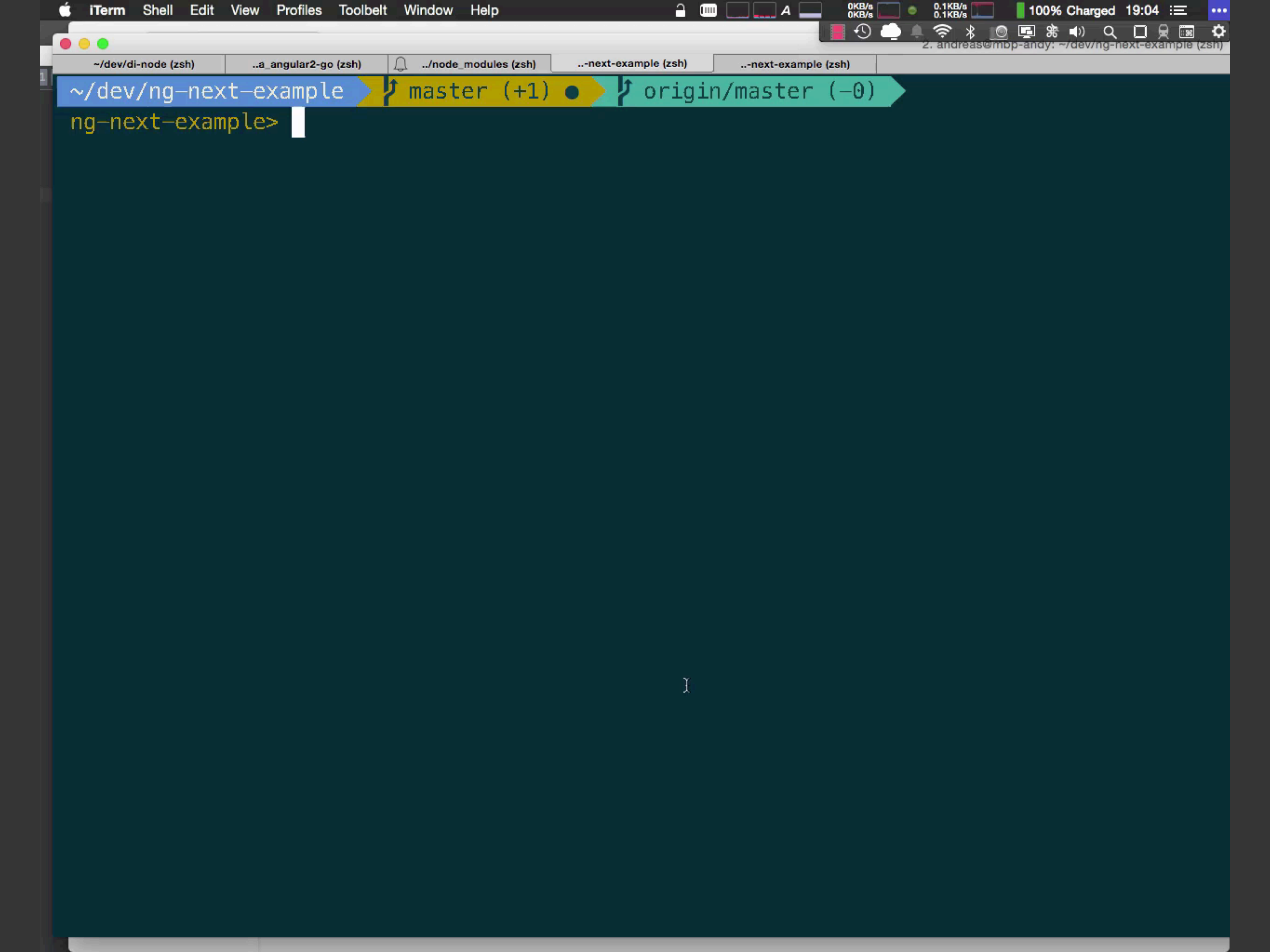
Firefox	Chrome	IE	Safari
39 ⊞ XP	43 🍎 10.10	11 ⊞ 8.1	8 🍎 10.10
39 🐧	43 ⊞ XP		
38 🍎 10.10	43 🐧		

ng-next

angular jspm material-design example

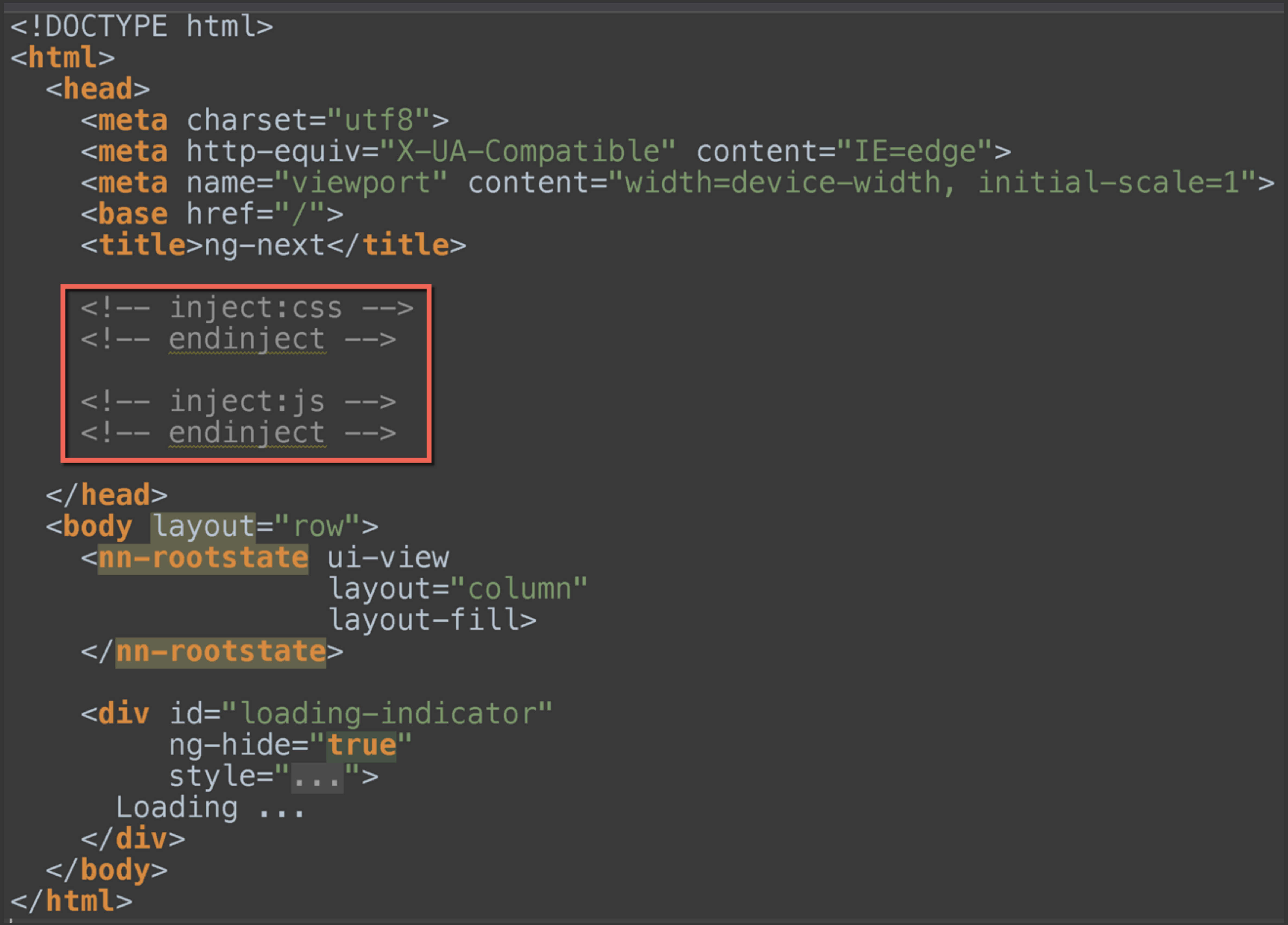
Run all the Gulp Tasks!

Sequentially / In Parallel / Respect Dependencies



```
~/dev/ng-next-example  master (+1) ●  origin/master (-0)
ng-next-example>
```

index.html

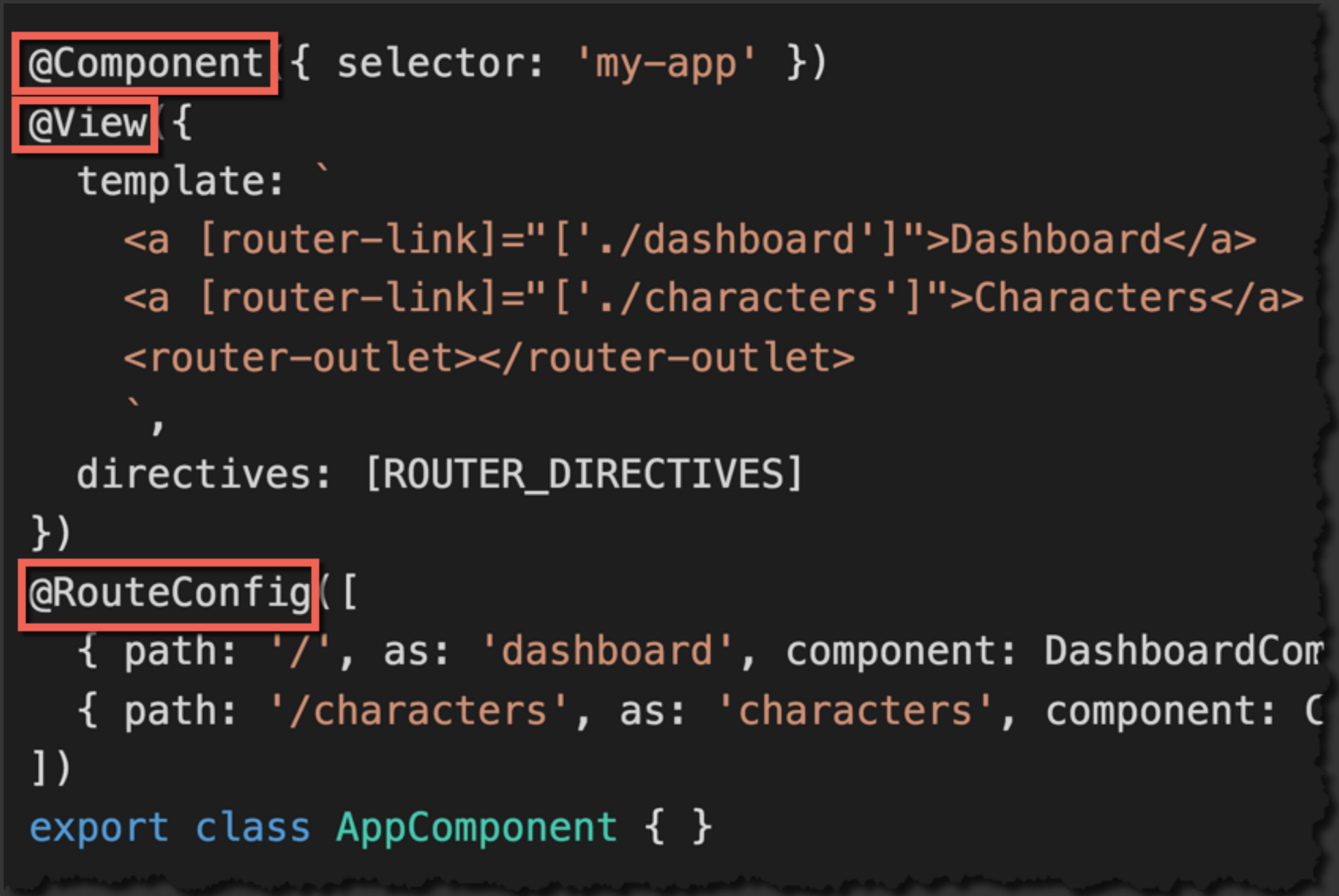


```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <base href="/">
    <title>ng-next</title>

    <!-- inject:css -->
    <!-- endinject -->

    <!-- inject:js -->
    <!-- endinject -->

  </head>
  <body layout="row">
    <nn-rootstate ui-view
                  layout="column"
                  layout-fill>
    </nn-rootstate>

    <div id="loading-indicator"
         ng-hide="true"
         style="...">
      Loading ...
    </div>
  </body>
</html>
```

SOLID JavaScript

Decorators

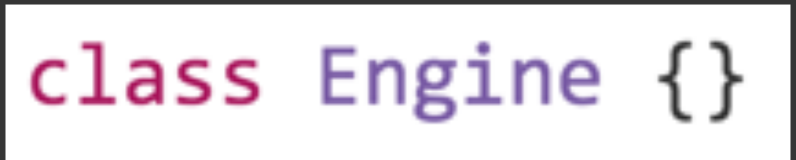
(probably) coming in ES7/ES2016



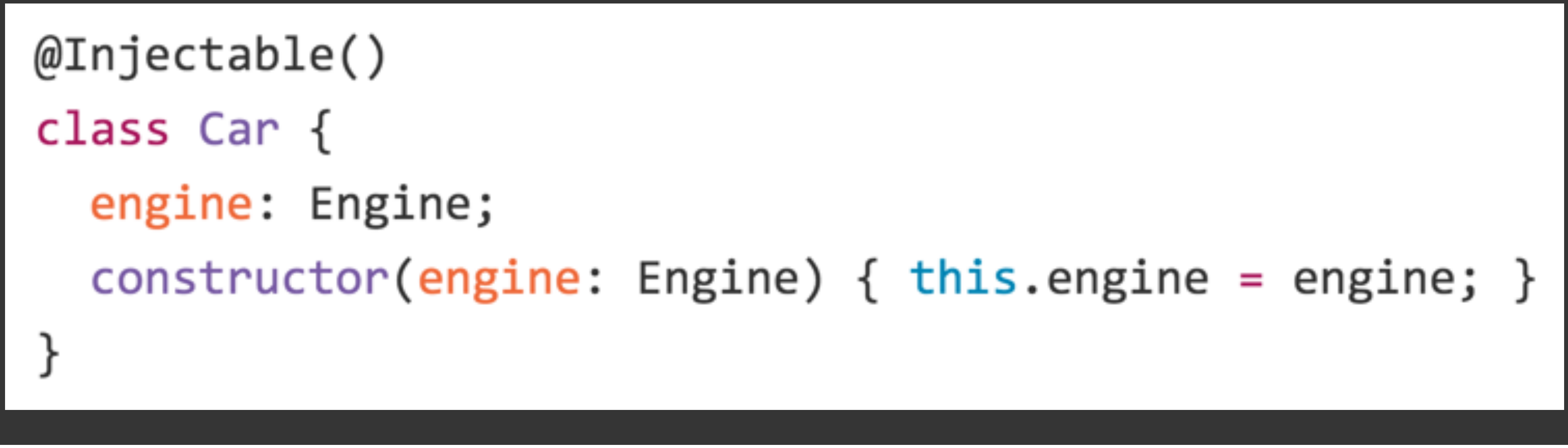
```
@Component({ selector: 'my-app' })
@View({
  template: `
    <a [router-link]="['./dashboard']">Dashboard</a>
    <a [router-link]="['./characters']">Characters</a>
    <router-outlet></router-outlet>
    `,
  directives: [ROUTER_DIRECTIVES]
})
@RouteConfig([
  { path: '/', as: 'dashboard', component: DashboardCom
  { path: '/characters', as: 'characters', component: C
])
export class AppComponent { }
```

Dependency Inversion Principle

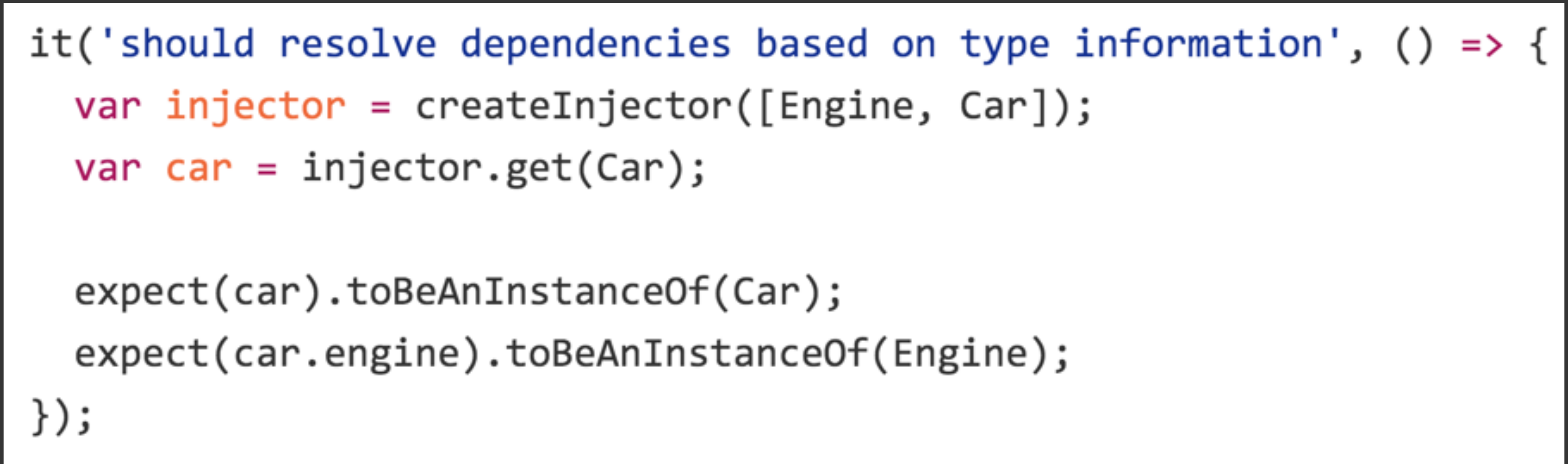
Angular 2



```
class Engine {}
```

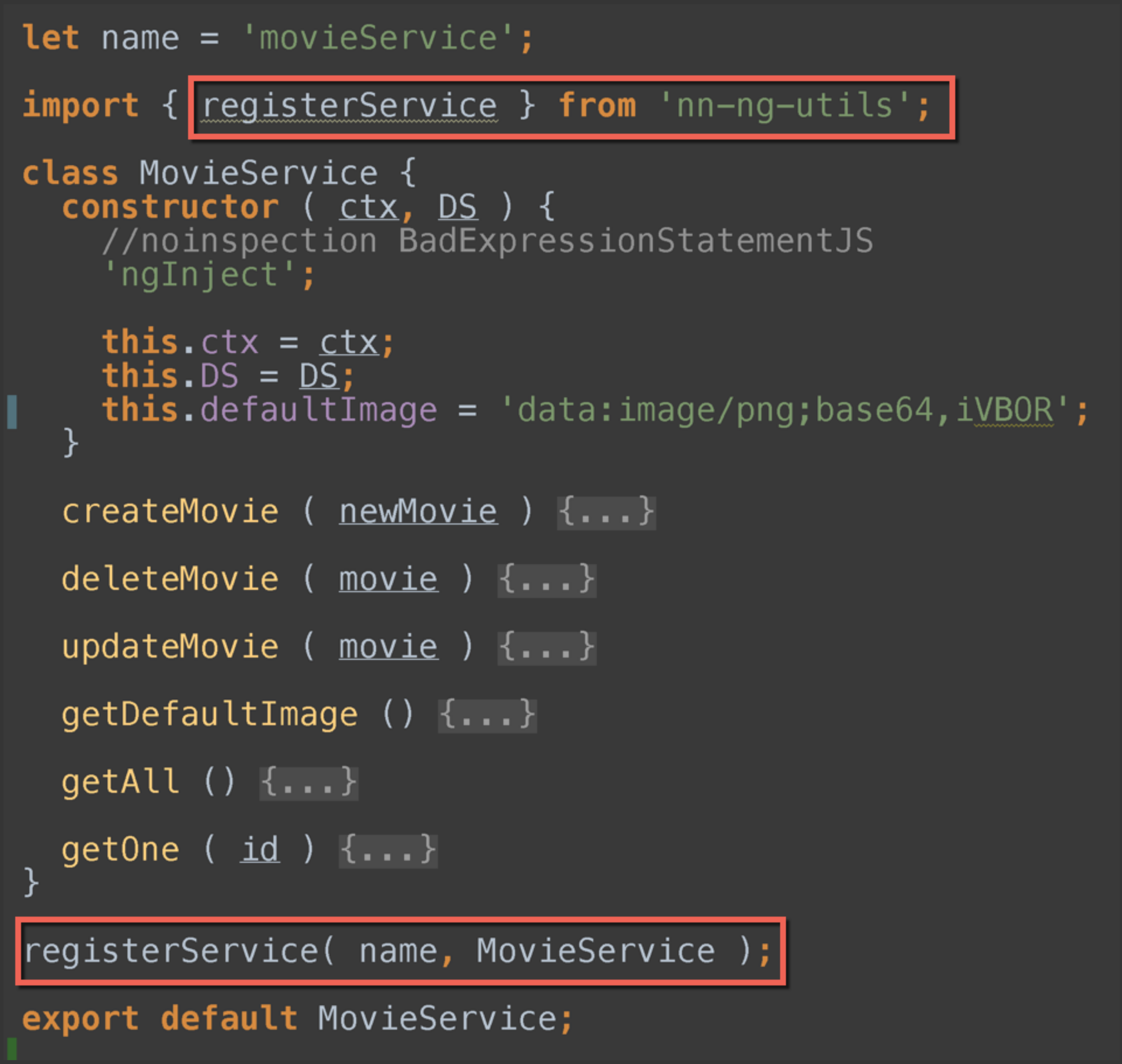


```
@Injectable()
class Car {
  engine: Engine;
  constructor(engine: Engine) { this.engine = engine; }
}
```



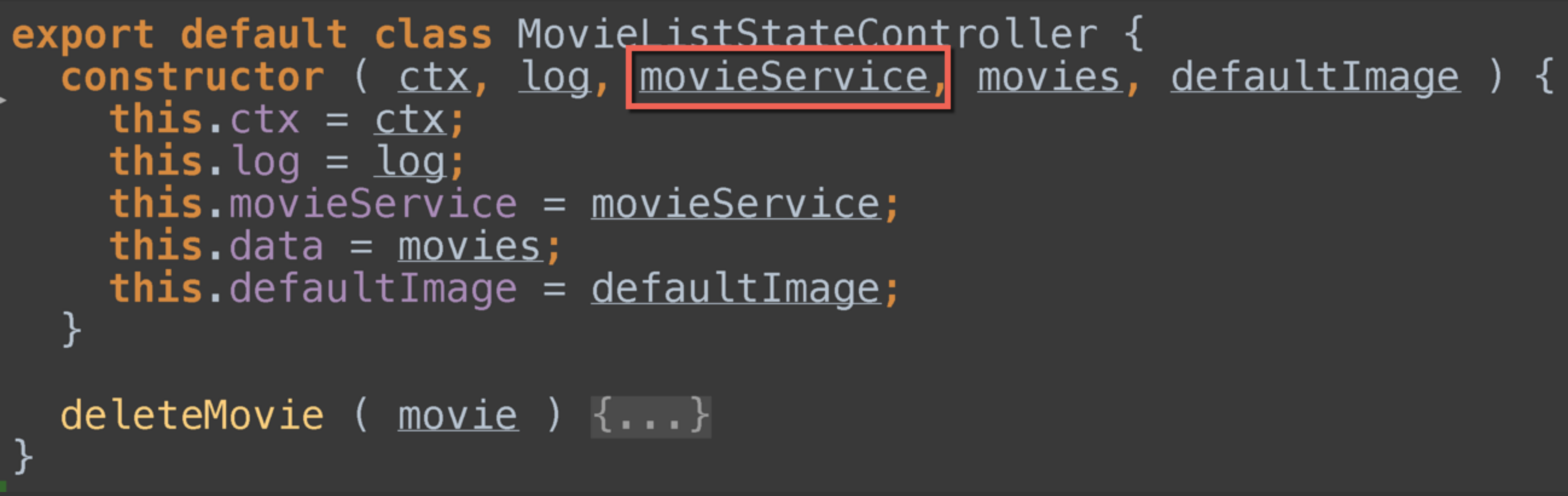
```
it('should resolve dependencies based on type information', () => {
  var injector = createInjector([Engine, Car]);
  var car = injector.get(Car);

  expect(car).toBeAnInstanceOf(Car);
  expect(car.engine).toBeAnInstanceOf(Engine);
});
```

Angular 1



```
let name = 'movieService';

import { registerService } from 'nn-ng-utils';

class MovieService {
  constructor ( ctx, DS ) {
    //noinspection BadExpressionStatementJS
    'ngInject';

    this.ctx = ctx;
    this.DS = DS;
    this.defaultImage = 'data:image/png;base64,iVBOR';
  }

  createMovie ( newMovie ) {...}

  deleteMovie ( movie ) {...}

  updateMovie ( movie ) {...}

  getDefaultImage () {...}

  getAll () {...}

  getOne ( id ) {...}
}

registerService( name, MovieService );

export default MovieService;
```

Angular 1



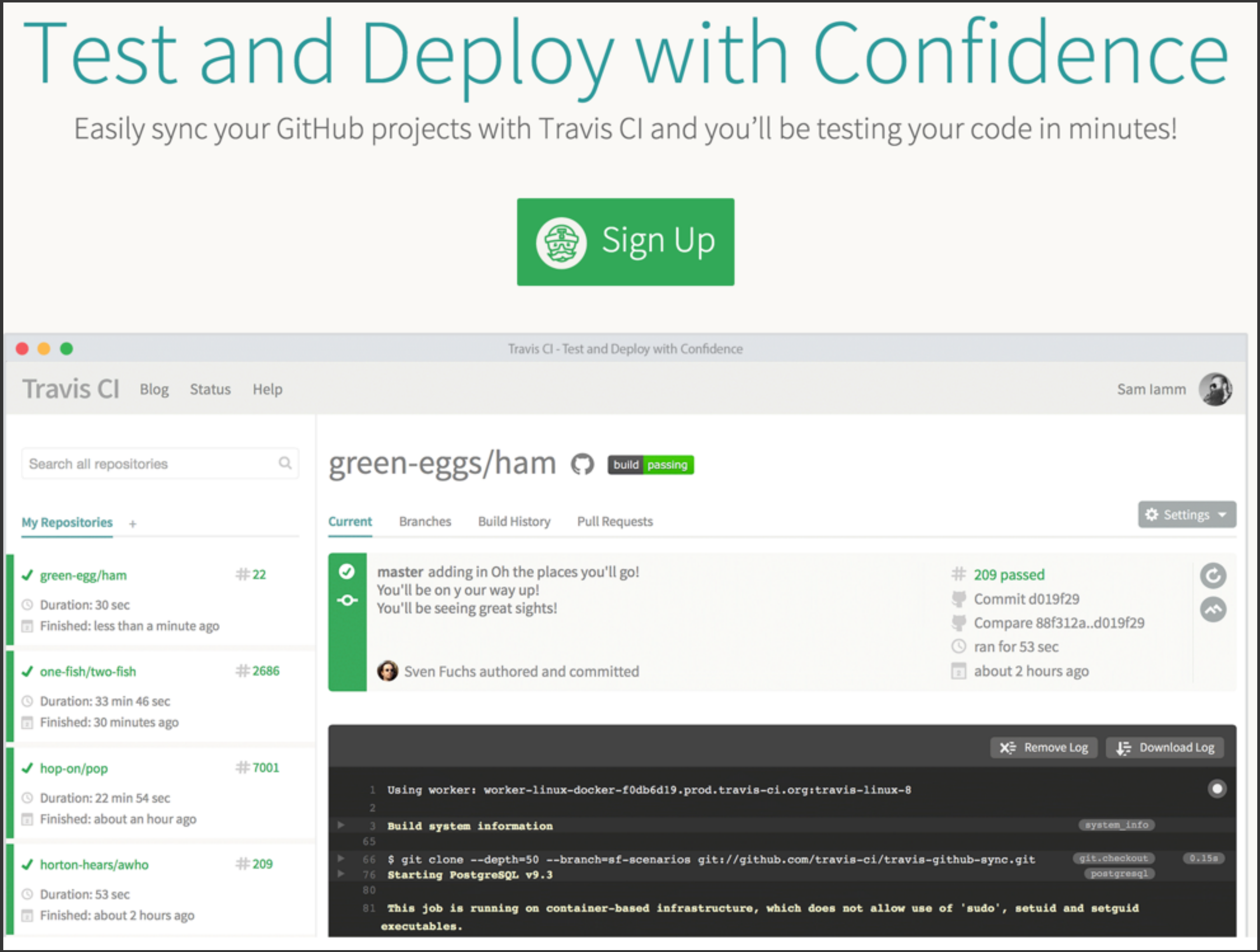
```
export default class MovieListStateController {
  constructor ( ctx, log, movieService, movies, defaultImage ) {
    this.ctx = ctx;
    this.log = log;
    this.movieService = movieService;
    this.data = movies;
    this.defaultImage = defaultImage;
  }

  deleteMovie ( movie ) {...}
}
```

Decoupling

"Dependency Free" Domain Objects / Entities

Continuous Integration

Travis CI



Test and Deploy with Confidence

Easily sync your GitHub projects with Travis CI and you'll be testing your code in minutes!

✓ green-egg/ham #22
Duration: 30 sec
Finished: less than a minute ago

✓ one-fish/two-fish #2686
Duration: 33 min 46 sec
Finished: 30 minutes ago

✓ hop-on/pop #7001
Duration: 22 min 54 sec
Finished: about an hour ago

✓ horton-hears/awho #209
Duration: 53 sec
Finished: about 2 hours ago

green-eggs/ham build passing

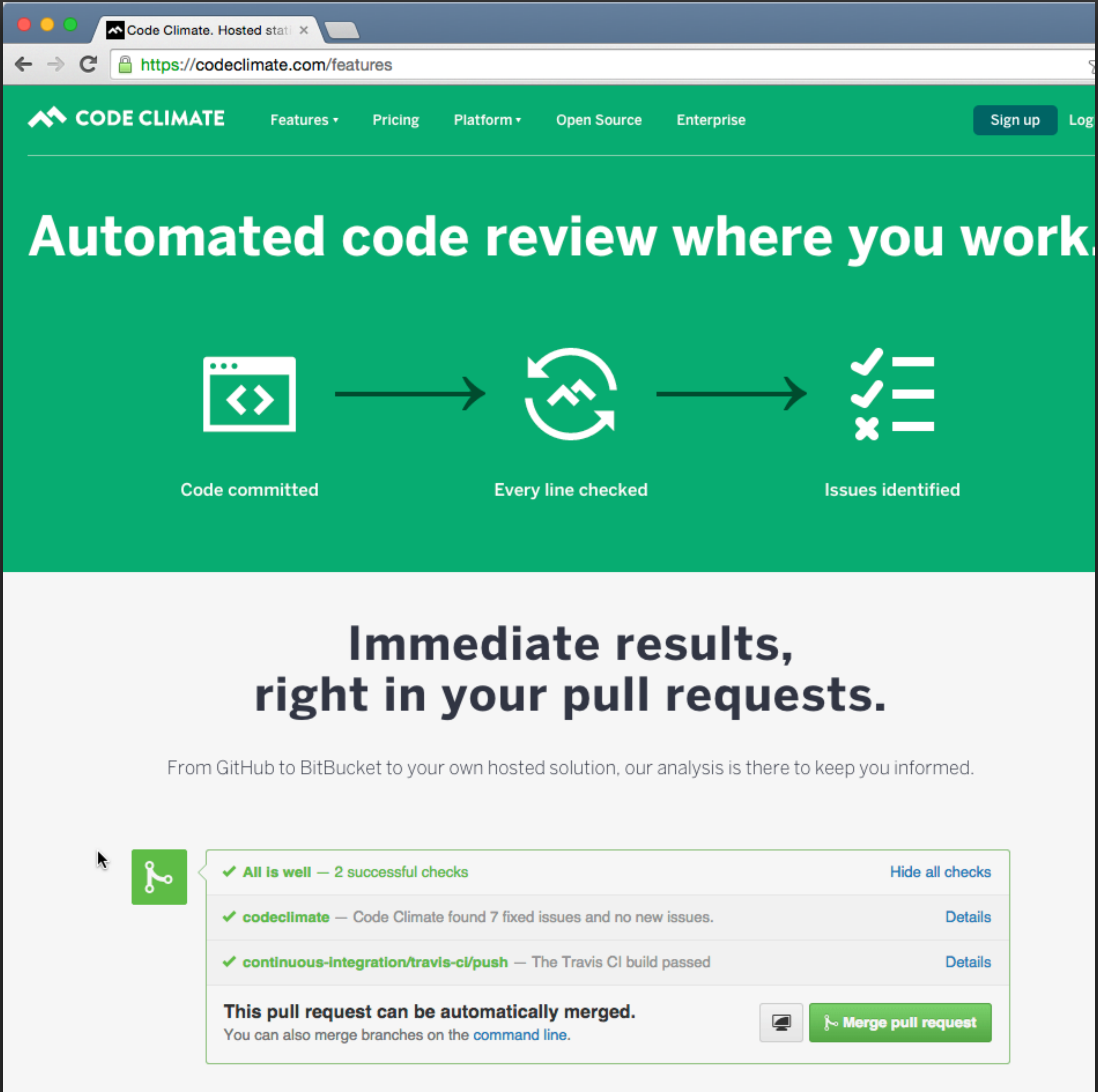
master adding in Oh the places you'll go!
You'll be on y our way up!
You'll be seeing great sights!

Sven Fuchs authored and committed

209 passed
Commit d019f29
Compare 88f312a..d019f29
ran for 53 sec
about 2 hours ago

```
Using worker: worker-linux-docker-f0db6d19.prod.travis-ci.org:travis-linux-8

Build system information                                                          system_info

$ git clone --depth=50 --branch=sf-scenarios git://github.com/travis-ci/travis-github-sync.git   git.checkout   0.15s
Starting PostgreSQL v9.3                                                          postgresql

This job is running on container-based infrastructure, which does not allow use of 'sudo', setuid and setguid
executables.
```

More Static Analysis



Automated code review where you work.

Immediate results,
right in your pull requests.

From GitHub to BitBucket to your own hosted solution, our analysis is there to keep you informed.

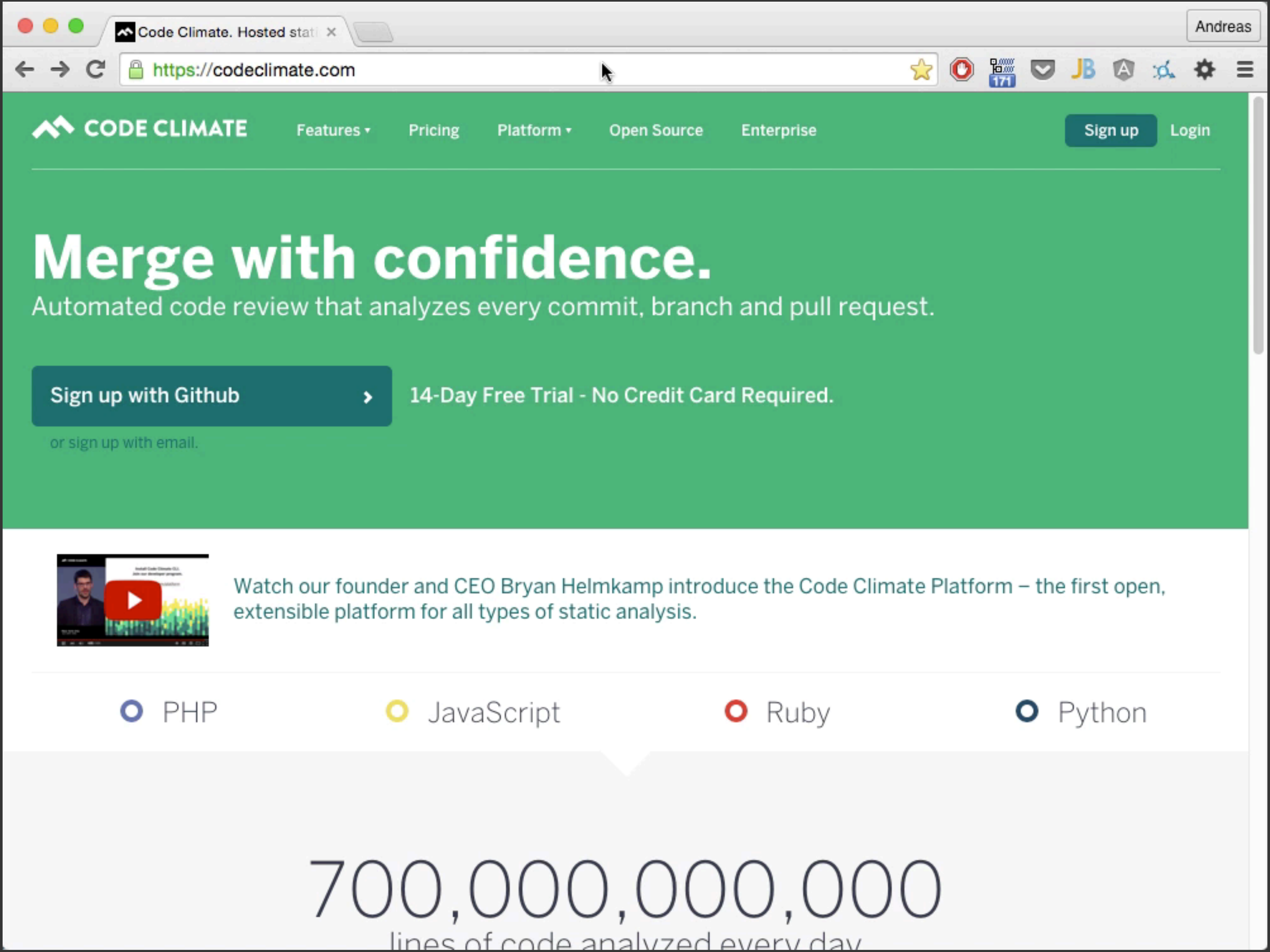
✓ All is well — 2 successful checks

✓ codeclimate — Code Climate found 7 fixed issues and no new issues.

✓ continuous-integration/travis-ci/push — The Travis CI build passed

This pull request can be automatically merged.
You can also merge branches on the command line.

https://codeclimate.com



Merge with confidence.

Automated code review that analyzes every commit, branch and pull request.

14-Day Free Trial - No Credit Card Required.

or sign up with email.

Watch our founder and CEO Bryan Helmkamp introduce the Code Climate Platform – the first open, extensible platform for all types of static analysis.

700,000,000,000

lines of code analyzed every day

We are the ones who know

Build the right thing!

Thank You!

- @_AndreasGrimm
- schaeferitsolutions@gmail.com
- github: andreasgrimm