Karma

Victor Herrero Cazurro

Contenidos

1. Introduccion

Test runner que permite ejecutar los test según se van codificando, obteniendo un feedback inmediato.

Permite trabajar con los siguientes frameworks de testing

- jasmine
- mocha
- qunit
- nodeunit
- nunit

2. Instalacion

Se recomienda inicializar un proyecto **Node.js** lanzando el comando

```
> npm init
```

Obteniendo así el fichero **package.json**

Se ha de lazar el siguiente comando para instalar **karma**

```
> npm install karma
```

Por defecto trabaja con **jasmine** como framework de pruebas, por lo que es necesario instalarlo

```
> npm install jasmine
```

E instalar el plugin de karma para jasmine

```
> npm install karma-jasmine
```

Para crear un proyecto **Karma**, se ha delanzar el comando

```
> node node_modules/karma/bin/karma init
```

Que creará el fichero de configuracion **karma.conf.js**, que tendra este aspecto

```
// Karma configuration
// Generated on Fri Oct 26 2018 07:20:34 GMT+0200 (Hora de verano
romance)

module.exports = function(config) {
  config.set({

    // base path that will be used to resolve all patterns (eg. files,
exclude)
    basePath: '',

    // frameworks to use
    // available frameworks: https://npmjs.org/browse/keyword/karma-
adapter
    frameworks: ['jasmine'],

    // list of files / patterns to load in the browser
    files: ['src/js/**/*.js', './test/**/*Spec.js'
    ],

    // list of files / patterns to exclude
    exclude: [
    ],

    // preprocess matching files before serving them to the browser
    // available preprocessors: https://npmjs.org/browse/keyword/karma-
preprocessor
    preprocessors: {
    },

    // test results reporter to use
    // possible values: 'dots', 'progress'
    // available reporters: https://npmjs.org/browse/keyword/karma-
reporter
    reporters: ['progress', 'html'],

    // web server port
    port: 9876,
```

```
    // enable / disable colors in the output (reporters and logs)
    colors: true,

    // level of logging
    // possible values: config.LOG_DISABLE || config.LOG_ERROR ||
config.LOG_WARN || config.LOG_INFO || config.LOG_DEBUG
    logLevel: config.LOG_INFO,

    // enable / disable watching file and executing tests whenever any
file changes
    autoWatch: true,

    // start these browsers
    // available browser launchers:
https://npmjs.org/browse/keyword/karma-launcher
    browsers: ['Chrome', 'PhantomJS'],

    // Continuous Integration mode
    // if true, Karma captures browsers, runs the tests and exits
    singleRun: false,

    // Concurrency level
    // how many browser should be started simultaneous
    concurrency: Infinity
  })
}
```

Los parametros mas interesantes de la configuracion son

- frameworks: Que define que frameworks de pruebas de van a interpretar.
- files: Que permite definir patrones de nombrado de los ficheros a cargar para ejecutar las pruebas
- preprocessors: Permiten definir tareas a realizar antes de procesar los test, uno de los usos que se le da es el de la cobertura
- reporters: Que permite definir como obtener el resultado, por defecto, se empleará **progress** que es la consola.
- autowatch: Que permite indicar a **karma** qeu inspeccione los ficheros de test en busca de cambios
- singleRun: Permite indicar a **karma** que despues de ejecutar los test termine el proceso.

Para arrancar la herramienta, se ejecutará el comando

```
> node node_modules/karma/bin/karma start
```

Que por defecto se ejecutará en **http://localhost:9876/**

2.1. Node.js

Hay que tener en cuenta que al cargarse en un navegador, no se puede hacer uso directamente se las funciones de **Node.js** como **require** o **module.exports**

```
//var Player = require('../src/Player');
//module.exports = Player;
```

2.2. Navegadores

Si se desea emplear otro navegador para la ejecucion de las pruebas, se deberá instalar el plugin correspondiente

```
//chrome
> npm install karma-chrome-launcher
// phantom
> npm install karma-phantomjs-launcher
// firefox
> npm install karma-firefox-launcher
// safari
> npm install karma-safari-launcher
// opera
> npm install karma-opera-launcher
// explorer
> npm install karma-ie-launcher
```

2.3. Reportes

Si se desean añadir otros reportes, se deberá instalar el plugin correspondiente

```
// HTML
> npm install karma-html-reporter
//JUNIT
> npm install karma-junit-reporter
```

Y añadir a la configuracion el uso del reporte

```
module.exports = function(config) {
  config.set({
    //. . . . .
    reporters: ['progress', 'html', 'junit'],
    //. . . . .
  })
}
```

2.4. PreProcesadores

Para configurar el calculo de la cobertura, se deberá hacer uso del plugin correspondiente

```
> npm install karma karma-coverage
```

Y configurar **karma** para que lo use

```
module.exports = function(config) {
  config.set({
    files: [
      'src/**/*.js',
      'test/**/*.js'
    ],

    // coverage reporter generates the coverage
    reporters: ['progress', 'coverage'],

    preprocessors: {
      // source files, that you wanna generate coverage for
      // do not include tests or libraries
      // (these files will be instrumented by Istanbul)
      'src/**/*.js': ['coverage']
    },

    // optionally, configure the reporter
    coverageReporter: {
      type : 'html',
      dir : 'coverage/'
    }
  });
};
```

2.5. SCM

Al repositorio de fuentes, se subiran unicamente

- test/
- src/
- karma.conf.js
- package-lock.json
- package.json

Al descargarlo, se debera ejecutar el comando

```
> npm update
```

Que descargará todas las dependencias de **Node.js** a directorio **node_modules** y a

partir de ese momento se emplea el proyecto de la misma forma que si se hubiera creado.

2.6. Jenkins

Para integrar en Jenkins este framework, basa con lanzar los mismo comandos a traves de un pipeline, dado que el comando **karma start** termina con un **exit code 1** cuando hay fallos en los test, se debe incluir la tarea **junit**, que muestra en jenkins la evolucion de los test, a través de un bloque **post** para asegurar que se ejecuta siempre, haya habido o no problemas.

```
pipeline {
  agent any
  stages {
    stage('Build') {
      steps {
        echo 'Se descarga todas las dependencias'
        bat 'npm update'
      }
    }
    stage('Test') {
      steps {
        echo 'Se ejecutan los test'
        bat 'node node_modules/karma/bin/karma start'
      }
    }
  }
  post {
    always {
       echo 'Se interpretan los resultados de los test'
       junit 'TESTS*.xml'
    }
  }
}
```