How to do ETE test with protractor in Configurator

# What is protractor?

Protractor is an end to end test framework for [AngularJS](http://angularjs.org/) applications built on top of [WebDriverJS](https://code.google.com/p/selenium/wiki/WebDriverJs). Protractor runs tests against your application running in a real browser, interacting with it as a user would, in other words is simulate user interactions.

# How to set up?

1. It needs Node.js environment.

<http://nodejs.org/>

1. Install Protractor (or omit the -g if you'd prefer not to install globally):

**npm install -g protractor**

1. Download and install the standalone server. Run

**webdriver-manager update**

1. Start the server:

**webdriver-manager start**

1. Start to test:

**protractor …(paths)/config.js**

# How to configure?

The configuration file tells Protractor what tests to run, how to connect to a webdriver server, and various other options for reporting. For an example:

**// An example configuration file.**

**exports.config = {**

**// The address of a running selenium server.**

**seleniumAddress: 'http://localhost:4444/wd/hub',**

**// The address of a running selenium server, specify for IE**

**// seleniumAddress: 'http://localhost:5555',**

**// Capabilities to be passed to the webdriver instance.**

**capabilities: {**

**'browserName': 'firefox'**

**// 'browserName': 'chrome'**

**// 'browserName': 'IE'**

**},**

**// Spec patterns are relative to the current working directly when**

**// protractor is called.**

**specs: [**

**// 'login/login.js',**

**'home/home-specs.js',**

**'logout/logout.js'**

**],**

**onPrepare: function() {**

**// The require statement must be down here, since jasmine-reporters**

**// needs jasmine to be in the global and protractor does not guarantee**

**// this until inside the onPrepare function.**

**require('jasmine-reporters');**

**jasmine.getEnv().addReporter(new jasmine.JUnitXmlReporter('test/reports/', true, true));**

**// login action module**

**var ptor = protractor.getInstance();**

**ptor.get('http://localhost:6080/config/#/login');**

**browser.driver.wait(function() {**

**return browser.driver.getCurrentUrl().then(function(url) {**

**return /http:\/\/localhost:6080\/config\/#\/login/.test(url);**

**})**

**});**

**var button = ptor.findElement(protractor.By.className('login-submit'));**

**element(by.model('data.userName')).sendKeys('admin');**

**element(by.model('data.password')).sendKeys('admin');**

**button.click();**

**browser.driver.wait(function() {**

**return browser.driver.getCurrentUrl().then(function(url) {**

**return /http:\/\/localhost:6080\/config\/#\/configuration/.test(url);**

**})**

**});**

**},**

**// Options to be passed to Jasmine-node.**

**jasmineNodeOpts: {**

**showColors: true,**

**defaultTimeoutInterval: 30000**

**}**

**};**

# How to write test case?

By default, Protractor uses [Jasmine](http://pivotal.github.io/jasmine/) as its test scaffolding. ([read about using mocha instead](https://github.com/angular/protractor/tree/master/docs/using-mocha.md)) Protractor exposes several global variables.

* browser this is the a wrapper around an instance of webdriver. Used for navigation and page-wide information.
* element is a helper function for finding and interacting with elements on the page you are testing.
* by is a collection of element locator strategies. For example, elements can be found by CSS selector, by ID, or by the attribute they are bound to with ng-model.
* protractor is the protractor namespace which wraps the webdriver namespace. This contains static variables and classes, such as protractor.Key which enumerates the codes for special keybord signals.

For an example of configurator, Login page:

**describe('MDM configurator ETE test: ', function() {**

**// predefine utils and ptor**

**var utils = require('../utils.js').utils;**

**var ptor = protractor.getInstance();**

**// test login action**

**describe('configurator login action', function() {**

**var targetUrl = 'http://localhost:6080/config/#/login';**

**ptor.get(targetUrl);**

**utils.waitUrl(new RegExp(targetUrl)).then(function() {**

**console.log("it goto the " + targetUrl);**

**});**

**it('should be login', function() {**

**var button = ptor.findElement(protractor.By.className('login-submit'));**

**targetUrl = 'http://localhost:6080/config/#/configuration';**

**element(by.model('data.userName')).sendKeys('admin');**

**element(by.model('data.password')).sendKeys('admin');**

**button.click();**

**utils.waitUrl(new RegExp(targetUrl)).then(function() {**

**console.log("it goto the " + targetUrl);**

**});**

**expect(ptor.driver.getCurrentUrl()).toEqual('http://localhost:6080/config/#/configuration');**

**}, 5000);**

**});**

**});**

# How to check the result?

1. Checked reports on command line window.
2. Configure in configuration file, then will generate an xml file contains report log. for an example:

**// An example configuration file.**

**exports.config = {**

**…**

**onPrepare: function() {**

**// The require statement must be down here, since jasmine-reporters**

**// needs jasmine to be in the global and protractor does not guarantee**

**// this until inside the onPrepare function.**

**require('jasmine-reporters');**

**jasmine.getEnv().addReporter(new jasmine.JUnitXmlReporter('test/reports/', true, true));**

**},**

**…**

**};**

# How to deploy protractor into grunt:

As we want to make an auto test framework, so it’s better to deploy test process into grunt.

1. It needs grunt environment, for details: <http://gruntjs.com/getting-started>
2. Install a grunt plugin for running protractor runner:

**npm install grunt-protrator-runner**

1. Configure the gruntfile.js

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

**// Project configuration.**

**grunt.initConfig({**

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

**…**

**protractor: {**

**options: {**

**configFile: 'node\_modules/protractor/referenceConf.js',**

**keepAlive: true,**

**noColor: false,**

**args: {**

**}**

**},**

**your\_target: {**

**options: {**

**configFile: 'test/ete/entry.js',**

**args: {}**

**}**

**}**

**}**

**});**

**…**

**grunt.loadNpmTasks('grunt-protractor-runner');**

**…**

**// grunt.registerTask('default', ['concat', 'jasmine', 'protractor']);**

**grunt.registerTask('default', ['protractor']);**

**};**

# Related website link:

The content above is brief, for more detail stuff, need to read the related website documents.

1. Node.js: <http://nodejs.org/>
2. Protractor: <https://github.com/angular/protractor>
3. Set up environment for Protractor: <https://github.com/angular/protractor/blob/master/docs/getting-started.md>
4. Jasmine: <http://pivotal.github.io/jasmine/>
5. Grunt: <http://gruntjs.com/>
6. Plug-in for deploy protractor to grunt: <https://www.npmjs.org/package/grunt-protractor-runner>

**Q&A**

If you have any questions or suggestions, please contact to Noah, the info:

1. Mail: [xtang@tibco-support.com](mailto:xtang@tibco-support.com)
2. Yahoo: [tang\_xuezhi@yahoo.cn](mailto:tang_xuezhi@yahoo.cn)