自动化测试工具

Wednesday, April 19, 2017 1:58 PM

1. Karma 一个工具：可以让你在多个浏览器中执行测试代码，它会启动一个httpserver，并生成一个用于测试的html文件，然后运行对应的测试框架 https://github.com/karma-runner/karma
2. Jasmine BDD测试框架, 它集expect，断言，test double等于一身 https://jasmine.github.io/edge/introduction.html
3. Mocha 测试框架，但是它只包含基本的功能，需要搭配断言、及多次测试库 http://mochajs.org/
4. Ava 号称下一代的测试框架，它同jasmine一样，也是集各种功能于一身，它与jsmine与mocha使用时的区别还是很大的 https://github.com/avajs/ava
5. Chai 断言库 http://chaijs.com/

5. Chai 断言库 http://chaijs.com/
6. Simon 多次测试库 http://sinonjs.org/#get-started
7. Cucumber 自动化测试框架，支持各种语言， 提供自然语言的支持，它可以和selenium结合使用
   https://cucumber.io/docs
   https://github.com/cucumber/cucumber-js

We can express the example as the following Scenario:

```
Scenario: Finding some cheese
  Given I am on the Google search page
  When I search for "Cheese!"
  Then the page title should start with "cheese"
```

Here are the accompanying Step Definitions:

```java
package com.example;

public class ExampleSteps {
    private final WebDriver driver = new FirefoxDriver();

    @Given("^I am on the Google search page$")
    public void I_visit_google() {
        driver.get("https://www.google.com");
    }

    @When("^I search for \"(.*)\"$")
    public void search_for(String query) {
        WebElement element = browser.findElement(By.name("q"));
```

8. Selenium 可以驱动浏览器，E2E测试利器，它有很多定位元素的方法，不过环境较复杂，它还可以做hybrid应用的测试

```
public class Selenium2Example  {
    public static void main(String[] args) {
        // Create a new instance of the Firefox driver
        // Notice that the remainder of the code relies on the interface,
```

```
        // not the implementation.
        WebDriver driver = new FirefoxDriver();

        // And now use this to visit Google
        driver.get("http://www.google.com");
        // Alternatively the same thing can be done like this
        // driver.navigate().to("http://www.google.com");

        // Find the text input element by its name
        WebElement element = driver.findElement(By.name("q"));

        // Enter something to search for
        element.sendKeys("Cheese!");

        // Now submit the form. WebDriver will find the form for us from the element
        element.submit();

        // Check the title of the page
        System.out.println("Page title is: " + driver.getTitle());

        // Google's search is rendered dynamically with JavaScript.
        // Wait for the page to load, timeout after 10 seconds
        (new WebDriverWait(driver, 10)).until(new ExpectedCondition<Boolean>() {
            public Boolean apply(WebDriver d) {
                return d.getTitle().toLowerCase().startsWith("cheese!");
            }
        });

        // Should see: "cheese! - Google Search"
        System.out.println("Page title is: " + driver.getTitle());

        //Close the browser
        driver.quit();
    }
}
```

9. Nightwatch.js 浏览器自动化测试，它使用了selenium driver http://nightwatchjs.org/
10. Enzyme react的测试实用工具，用它来mount一个组件 https://github.com/airbnb/enzyme