Functional Testing

Devdatta Kulkarni

Functional Testing

- Testing from end user’s perspective whether the REST Service/API is working as expected
- Steps:
  - Design and implement your REST Service
  - Create tests that programmatically exercise the API
  - For each test:
    - Save the response
    - Write assertions

Differences between unit and functional testing

- Unit testing is concerned with testing individual layers of your application
- Functional testing is concerned with testing whether the entire application is working as expected or not
- Unit testing involves mocking out the dependencies of the code under test
- Functional tests do not involve mocking

Differences between unit and functional testing

- Changing application’s internal architecture should not affect functional tests if they are written properly

- Unit tests are *supposed to* get affected by changes to application’s architecture

Similarities between unit testing and functional testing

- Both involve writing assertions on the output

Functional testing tools

- What does such a tool need?
  - Ability to make REST calls
  - Ability to capture response data
  - Ability to parse response data using different parsing criteria
  - Ability to write and verify assertions
- SOAPUI
  - http://www.soapui.org/
- Spock
  - https://code.google.com/p/spock/
- Straight Java libraries

SOAP UI

- UI-based test tool
- Supports using of XPath for parsing response data via language Groovy
- Supports sharing test code between different test plans
- We will *not* use it
  - Latest version seems unstable (at least on Mac)
  - Need to learn Groovy

Spock

- https://code.google.com/p/spock/
- Java and Scala based functional testing tool
- Requires understanding of Java and Scala
- We will not use it

Approach that we will use

- Use Java code
- Example:
  - ex03_1
- Conceptual steps:
  - Build a Client
  - Make a request
  - Read the response
  - Assert

Running Functional Test

- Orielly Book example: ex03_1
- Steps:
  - New -> Java Project
  - Uncheck “Use default location”
  - Browse to the folder containing code for ex03_1
  - Change Junit pom dependency to use version 4.12
  - Next->Finish
  - Configure -> Convert to Maven Project
  - Maven -> Update Project
  - Change Context Root from ex03_1 to “/”
    - Right click on project
    - Properties
    - Web Project Settings
  - Deploy the application by running the server
  - Run “CustomerResourceTest”
  - The test should pass