Acceptance Test Driven Development with Cucumber & Java (Custom for Capital One)

Course Summary

Description

This is a two day overview course which introduces students to the ideas, practices and methodology of Acceptance Test Driven Development (ATDD).

The first day explores the collaborative practices used in the ATDD cycle of Discuss – Distill – Develop - Demo from initial requirements and specification by example through writing executable acceptance tests in Gherkin. Emphasis is on the integration of ATDD with other best practices such as continuous testing, software craftsmanship, Agile software delivery, test driven development, Agile testing and others while remaining focused on ATDD within the Capital One IT context. The application of ATDD in the areas of new application development, support programming in existing systems and API or micro-services are covered.

The second day focuses on implementing automated acceptance tests using Cucumber, Java, TestNG, Maven and other tools and frameworks used at Capital One. Students work through illustrative examples of converting Gherkin features into test automation code emphasizing the best practices in writing, testing, deploying maintaining the automation code. Automation of tests in web based applications and RESTful APIs are examined in detail.

Objectives
At the end of this course, students will be able to:

- Describe the ATDD Discuss – Distill – Develop – Demo cycle and the activities that occur at each stage.
- What the common errors are that occur during the software development process and how ATDD practices work to prevent or correct those errors.
- Explain and perform the collaborative activities used in each stage of ATDD.
- Create robust acceptance tests in Gherkin
- Implement Gherkin acceptance tests using Cucumber and related tools to implement executable acceptance tests for web based based applications, RESTful APIs and other interfaces.
- Tailor the ATDD concepts, practices and tools to fit their own specific environment and processes.
- Integrate ATDD with other best practices at Capital One such as software craftsmanship, TDD, Agile development and domain driven design.
- Develop domain specific language constructs used in writing acceptance tests.

Topics

- Root causes of failures and inefficiencies in software development and delivery.
- How ATDD is used to correct or prevent those errors that are the root causes of failure.
- The ATDD process: Discuss-Distill-Develop-Demo and how it integrates with Agile development.
- Discuss Activities: delivery of specifications by example and acceptance criteria.
- Distill Activities: development of lean and robust acceptance tests.
- Writing acceptance tests in Gherkin: declarative versus imperative.

Acceptance Test Driven Development with Cucumber & Java (Custom for Capital One)

Course Summary (cont'd)

- Gherkin best practices: using Scenario Outlines, domain specific language, Scenario and feature file organization.
- Testing the tests: ensuring the Gherkin acceptance tests are robust and effective.
- How Cucumber automates: step definitions.
- Implementing step definitions in Java.
- Using TestNG to run Cucumber Tests
- Selective execution of tests using tags.
- Implementing Step definitions for generic application interfaces.
- Implementing Step definitions for web applications.
- Using the Page Object pattern in web testing.
- Implementing step definitions for RESTful APIs.
- Using ATDD to support and integrate with Test Driven Development.
- Using Cucumber in Eclipse and batch environments.

Audience

This course is for anyone who needs to understand what ATDD is, how it is implemented at Capital One. It is directed towards those who have little or no exposure to ATDD and are working with it or will be working with it in some role at Capital One. This class is suitable for testers, developers, Scrum Masters, business analysts and product owners.

Prerequisites

None required for the first day, but the second assumes the ability to understand and write Java code at a basic level as well as familiarity with TestNG or a similar automated test environment like JUnit. The ability to use Eclipse as a Java IDE at a basic level is also assumed.

Duration

Two days