

**Acceptance Test Driven Development (ATDD)**

**Course Number:** JAV-308
**Duration:** 3 days

**Overview**

Accelebrate's ATDD training course teaches participants how to create acceptance tests that verify their correct implementation and accurately transform customer requirements into product specifications.

**Prerequisites**

All attendees should have prior Java development experience.

**Materials**

All attendees receive comprehensive courseware covering all topics in the course.

**Software Needed on Each Student PC**

* Windows, macOS, or Linux with at least 8 GB RAM
* A recent JDK version installed
* Related software that Accelebrate will specify in a detailed setup sheet following the purchase of this class

**Objectives**

* Capture the core JUnit syntax
* Use BDD and the test-fail-first approach
* Translate business requirements into user stories with acceptance tests
* Learn how to Refactor to clean up your code
* What makes your code testable
* Use Fakes and Mocks for isolated unit tests
* Use Cucumber-JVM to turn User Stores into Code
* Explore the Selenium API

**Outline**

* How Stories Fit Into the Agile Process
	+ What is Agile?
	+ The role of the product owner
	+ Criteria for evaluating stories
	+ Requirement analysis
	+ Use case modelling
	+ Herringbone diagram
	+ Context diagrams
	+ The role of the product backlog
* Behavior Driven Development (BDD) with Cucumber
	+ Writing user stories
	+ Acceptance criteria
	+ Specification by example
	+ Creating software based on realistic examples
	+ Bridging the communication gaps among business stakeholders
* Introduction to Cucumber
	+ Writing scenarios with Gherkin
	+ Gherkin syntax
	+ Using given, when, then
	+ Java, Ruby, or Groovy fixtures generated from scenarios
	+ Using JUnit to run your Cucumber stories and scenarios
	+ Writing set-up and tear down code using hooks (@Before and @After)
	+ Automate Cucumber steps to drive your application both through and below the user interface
	+ Refactor Cucumber step definitions to make them more readable and maintainable
* Selenium
	+ Web testing
	+ A history of selenium
	+ Selenium API – web driver
	+ Assertions
	+ Matching
	+ Waiting
	+ Storing
	+ JavaScript
	+ The Selenium IDE
	+ Building tests
	+ Running test suites
	+ Best practices
* Testing Web Sites with Selenium with Cucumber
	+ Using a WebDriver to test web applications and the Selenium API
	+ Interrogating a response page
	+ Simulating links and form submissions
	+ Simulate multiple page navigation
	+ Continuous integration
* Stories and Test Driven Development (TDD)
	+ Principles and techniques
	+ TDD metaphors
	+ Benefits, challenges and limitations
	+ Handling requirements change
	+ Characteristics of good tests
* Testable Designs (Mocks, Fakes and Stubs)
	+ Creating testable code, If you cannot test it what use is it?
	+ Stubs, Fakes and Mocks
	+ Mocks as collaborators
	+ Mocks and return values, void methods, frequency calls and ordering
	+ Cucumber, Mocks and all that!
* Conclusion