Implementing Agile Test Driven Development for Non-Developers

Course Summary

Description

Implementing Agile Test Driven Development for Non-Developers is a two day, in-depth Agile-testing training course that provides students with a solid introduction to Test Driven Development and "test first" design within the context of Agile processes and practices.

Test-driven development (TDD) is an evolutionary approach to development where you must first write a test that fails before you write new functional code. This process was developed by Kent Beck and Ward Cunningham. It is primarily an agile approach to software development and is one of the core principles of Extreme Programming.

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

- Be introduced to the concept of development agility and the Agile Manifesto
- Review each of the major agile development methods underscoring their strengths and weaknesses
- Understand how to manage an agile environment even within a structured organizational approach
- Learn how to introduce agility into a development organization
- Examine what unit testing is and how various xUnit frameworks facilitate unit testing
- Review the xUnit family of unit testing tools
- Understand the concepts of and motivations for Test-Driven Development
- Relate unit testing, test driven development, and test coverage to agile processes
- Understand the importance of refactoring in supporting agile and test driven processes
- Understand what Continuous Integration is and what the components of CI are
- Examine the motivations for CI
- Review best practices for everything from CI to testing within the context of agile development

Topics

- Agile Development
- Unit Testing
- Agile Testing Best Practices

Audience

This is a beginner- level course, designed for test professionals, test managers, project leaders, quality analysts, and developers.

Prerequisites

No specific prerequisites are assumed. Knowledge of current development processes, such as structured top-down development and the waterfall method is beneficial.

Duration

Two days

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

Implementing Agile Test Driven Development for Non-Developers

Course Outline

I. Agile Development
- A. Agile Rationale and Concepts
  1. Reducing Risk Through Agility
  2. The Discipline of Timeboxing
  3. Incremental Delivery and Evaluation
  4. Agile Method: Scrum
  5. Agile Method: XP
  6. Pair Programming
- B. The Agile Approach
  1. Agile Software Development Manifesto
  2. The Agile Principles
  3. Identifying Features
  4. Managing Features
  5. Communication Dynamics
- C. Agile Iterative Development
  1. Iterative Approaches
  2. Phased Iterative Development
  3. Iterating
  4. Feasibility & Planning
  5. Development
  6. Adaptation & Deployment
- D. Prioritizing and Planning
  1. Features and Backlogs
  2. FDD Process
  3. Prioritizing Features
  4. Release Planning
  5. Assigning Features to Iterations
- E. Building
  1. Typical Continuous Integration Process
  2. CI Server
  3. Automate Source Code Management
  4. Automate Build Process
  5. Automate Testing
  6. Automate Deployment

II. Unit Testing
- A. Unit Testing Overview
  1. Purpose of Unit Testing
  2. Good Unit Tests
  3. Test Stages
  4. Unit Testing Vs Integration Testing
- B. Unit Testing Tools
  1. Understanding Unit Testing Frameworks
  2. JUnit Overview
  3. Test Case using JUnit
  4. Failures vs. Errors
- C. Unit Testing Best Practices
  1. "Good" Tests
  2. Bad Smells
  3. White-Box Unit Testing
  4. Black-Box Unit Testing
  5. Automation and Coverage

III. Agile Testing Best Practices
- A. Transitioning to Agility
  1. Agility: Some Process, Some Mindset
  2. Characteristics that Enable Agility
  3. Characteristics that Inhibit Agility
  4. Risks Associated with Migrating
  5. Smoothing the Transition
- B. The Bottom Line
  1. Agile Migration Patterns
  2. Extending the Migration
  3. Coding Practices
  4. Source Control
  5. Pair Programming and Code Reviews
  6. Continuous Integration
  7. Legacy Cod

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically