Agile Development in Java (with Kevlin Henney)

Learn via: **Classroom / Virtual Classroom / Online**

Duration: **3 Day**

https://bilginc.com/en/training/agile-development-in-java-with-kevlin-henney-5592-training/

Overview

Agile principles, practices and processes offer a path to sustainable development for individuals, teams and organisations. For many developers who want to focus on their craft, however, it is sometimes difficult to get a view of Agile development that is not either focused on a project management perspective or just on the practice of Test-Driven Development (TDD).

For the Java developer, an overview of the larger Agile process landscape needs to be complemented with the practical side of software craftsmanship. This ranges from understanding how Scrum can be fine tuned with Lean thinking to exploring Extreme Programming practices, such as TDD and pairing.

The Agile Development in Java course is aimed at Java developers who want to learn what Agile means for them. It introduces a number of common agile techniques and puts these into practice in labs and exercises in pairs and groups, before applying these over a series of mini-iterations. The workshop balances taught material with practice, introducing requirement techniques, lightweight modeling techniques, tracking and estimating approaches, design principles, testing practices and refactorings.

About The Trainer

Kevlin Henney is a regular columnist for various industry magazines and a well known and popular speaker on topics such OO Design, Patterns, Agile Development and Software architecture at conferences in Europe and North America.

Kevlin currently works as an independent consultant and trainer based in Bristol. He has developed and delivered training courses, consultancy and software across a number of domains ever since getting involved in professional software development in the late 1980s.

Most of Kevlin's work focuses on software architecture, patterns, development process and programming languages. His work has appeared in several magazines and online publications, including; The Register, Application Development Advisor, Java Report, C++ Report and CUJ. Along with Frank Buschmann and Doug Schmidt, Kevlin is coauthor of A Pattern Language for Distributed Computing and On Patterns and Pattern Languages. He is also editor of the 97 Things Every Programmer Should Know project.

INSTIL

Prerequisites

The course is suitable for Java developers who wish to learn more about the practical side of Agile development, particularly TDD and incremental development.

What You Will Learn

- Describe representative agile development processes and common practices
- Slice up requirements in terms of goals and estimate and plan against them
- Understand design thinking appropriate for responsive development
- Learn how to carry out TDD effectively
- Put concepts and techniques into practice during labs and in a small, iterative workshop

Outline

Agile Development

- Software development and change

- Agile values and principles
- Iterative and incremental development
- Visualisation of progress
- Kicking off and closing out iterations
- The role of testing
- Modeling in an Agile context
- Plan–Do–Study–Act

Common Agile Approaches

- Extreme Programming
- XP1 and XP2
- Scrum
- Scrum roles, events and artefacts
- The Nokia test
- Lean Software Development
- Lean principles
- Kanban for software
- Limiting work in progress (WIP)

Software Craftsmanship

- Code quality and development skills
- Elements of well-crafted code
- Coding guidelines benefits and pitfalls
- Code sufficiency vs over design
- Technical debt and code smells
- Refactoring
- Programmer testing

Test-Driven Development

- Good Unit Tests (GUTs)
- Plain Ol' Unit Testing (POUT)
- Defect- Driven Testing (DDT)
- Test-Driven Development (TDD)
- Key TDD practices and the test-first cycle
- Behavioural testing based on propositions
- Negative test cases
- Overview of JUnit

Design Practice

- Agile architecture and responsive design
- Pattern thinking
- Class hierarchy design
- Acyclic dependencies
- Interface decoupling
- Transitive and external dependencies
- Test doubles
- Components with single responsibilities

Goal-Structured Requirements

- Utilising use cases, scenarios and user stories
- Incremental development
- Lightweight use cases
- User story styles and guidelines
- Prioritisation in terms of value and risk
- Scenario and task estimation
- Tracking