Clean Code

Duration: 5 Days
Available Languages: English German

Audience

Software Crafter: Software Developers, Architects, Project Managers, Scrum Masters.

Precondition

Good knowledge of an object-oriented programming language like Java.

Goals

Learn how to develop software using the Clean Code principles.

Contents

- The two values of Software
- Design Smells: Rigidity, Fragility, Immobility/Inseparability, Opacity, Viscosity
- Programming Paradigms
    - # Structured Programming
    - # Object-Oriented Programming, Polymorphism
    - # Functional Programming
    - # Predicate Logic
- Craft Culture, Professionalism, and Clean Code
- Clean Code Fundamentals: Comments, Meaningful Names, Clean Functions and Classes, Source Code Structure, Code Style, Formatting, Boundaries, Error Handling
- Error Handling
- Core Architecture: Coupling and Cohesion, Vertical (Domain) vs Horizontal (Technical) Software Architecture
- TDD - Test-Driven Development
    - # What's a Unit Test?
    - # Black Box vs Grey Box vs White Box
    - # The Three Laws of TDD
    - # Red-Green-Refactor Cycle
    - # TPP
    - # The 4 A's
    - # Single Assert Rule
    - # ATDD
    - # Ping Pong
    - # ZOMBIES
    - # Property Testing
    - # Mutation Testing
- BDD - Behavior Driven Development

    - # Gherkin
    - # The 4 A's in Gherkin
    - # Relationship between BDD and the Agile SDLC
- The Test Pyramid, layers of tests
- The Clean SDLC: Agile, Scrum, XP, CI/CD, DevOps
- Continuous All The Things
    - # Testing
    - # Refactoring
    - # Design Improvement
    - # Integration
    - # Release
    - # Delivery
    - # Deployment
    - # The DevOps Pyramid
- The SOLID and Package Principles: 11 Principles of Clean Architecture and Design
    - # SRP, OCP, LSP, ISP, DIP
    - # Package Principles and Package Metrics
- Design Patterns in the light of Clean Code
    - # Creational Patterns
    - # Structural Patterns
    - # Behavioral Patterns
- Catalogue of Code Smells: Alternative classes with divergent interfaces to temporary fields
- Catalogue of Refactoring Techniques: From Algorithm Substitution to Superclass Extraction
- Refactoring to Patterns: From Unifying interfaces through adapters to replacing constructors with factories
- Smells and Heuristics: Comments, Environment, Functions, General, Names, Tests
- Some Pragmatic Tips

The training can be offered in all major programming languages. For corporate training, please contact us in advance for planning the training to meet needs and environment of your development teams.

Booking

Contact Siddhesh Nikude, +91-95-52572354, training@nelkinda.com