Test-Driven Development for Embedded with AceUnit

Duration: 3 Days
Available Languages: English German

Audience

Software Craftsmen: Embedded Software Developers, Embedded Architects, Embedded Scrum Masters.

Precondition

Good knowledge of the C programming language.

Goals

Learn how to develop embedded software using Test-Driven Development.

Contents

- Software Architecture Fundamentals for TDD
    - # The two values of software
    - # The ATP-Trinity of project code
    - # The Importance Priorities: Automation > Test > Production
    - # The five major design smells
    - # Cohesion and Coupling
    - # What is "testable" and how is it related to maintainability / Clean Code?
    - # Test Automation Pyramid
    - # TDD in the context of Agile and XP
- Unit Testing fundamentals
    - # The job of a test framework
    - # Anatomy of xUnit frameworks
    - # AceUnit, CUnit, CppUnit, EmbUnit, Google Test
    - # The Single-Assert Rule
- TDD fundamentals
    - # The Three Laws of Test-Driven Development
    - # The Red-Green-Refactor Cycle
    - # The FIRST principles
    - # How to Start
- AceUnit
    - # Integrating AceUnit
    - # Configuring AceUnit
    - # Writing AceUnit tests
    - # Running AceUnit tests
    - # The AceUnit test results

- Test Doubles (Stubbing and Mocking)
  - # The Ontology of Test Doubles
  - # The Two Schools of TDD: Stateism vs Mockism
  - # Mocking Techniques in Embedded Software Development
- BDD - Behavior Driven Development
  - # Unit Testing vs Acceptance Testing
  - # Given-When-Then vs 4 A's
- TPP - Transformation Priority Premise
  - # TPP - Transformation Priority Premise
  - # Transformation vs Refactoring
  - # Starting Points
  - # The Sequence for Tests
  - # TCR - test && commit || revert
- ATDD - Acceptance Test-Driven Development
  - # Test Automation Pyramid
  - # Using BDD on different layers
  - # Test Step Definitions on the acceptance level
  - # Test Step Definitions on the integration level
  - # Test Step Definitions on the unit level
- TDD for Legacy Code
  - # Legacy Code Change Matrix
  - # Refactoring
  - # Characterization Testing
- Outlook
  - # Working Effectively with Legacy Code
  - # How to migrate Test-Last to Test-First
  - # TDD and the SOLID principles
  - # TDD and Agile Development (Scrum, XP, Kanban, Lean)
  - # TDD and Software Craftsmanship
  - # TDD and Pair Programming - Ping Pong
  - # TDD and Continuous Integration / Continuous Delivery / DevOps

The course uses C17 with the GNU C Compiler or Clang, and AceUnit. C2x preview features like attributes or the harmonization of static_assert with C++ will be covered briefly when relevant for testing and depending on feature support in the available compilers. Differences between AceUnit, CUnit, CppUnit, EmbUnit, Google Test are covered in detail.

The course language is C. Nelkinda also offers this course in other languages, for example, C++, C#, JavaScript, Kotlin, and Python.

Event Type

This is a full-day open (anyone can register) instructor-led classroom training about Test-Driven Development in Java. The number of seats is limited to ensure the best quality training for the participants. The course fee includes snacks and lunch.

Trainer

Your trainer for this event is Christian Hujer.

Christian Hujer has experience with embedded CPU and Microcontrollers since 1984, for example, Zilog Z80A, MOS 6502, Motorola, 68000, Samsung CalmRISC/SecuCalm, ARM, Infineon TriCore, Atmel AVR, Hitachi H8, and Intel 80x51. He has 20 years of experience in TDD. He's been training developers and teams for organizations like BNP Paribas, Elsevier, Giesecke & Devrient, Nokia, SUN Microsystems, Volkswagen, and many others.

Booking

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