L4 - Industrial Java

Developing Industrial Applications in Java(TM)

Java is a Registered Trade Mark of Oracle

Objectives

- Master the concepts of Java
- Secure your Java applications exception handling language
- Master Java threads
- Learn Applet creation and use
- Learn how to call C/C++ functions from Java programs through JNI
- Use collections of objects in Java
- Master the main utility classes in Java
- Optimize the Java code

Practical exercises may be done either locally on the PC or by using the MicroEJ SDK to target an embedded microcontroller

Course Environment

- Theoretical course
  - PDF course material (in English) supplemented by a printed version for face-to-face courses.
  - Online courses are dispensed using the Teams video-conferencing system.
  - The trainer answers trainees' questions during the training and provide technical and pedagogical assistance.
- Practical activities
  - Practical activities represent from 40% to 50% of course duration.
  - Code examples, exercises and solutions
  - For remote trainings:
    - One Online Linux PC per trainee for the practical activities.
    - The trainer has access to trainees' Online PCs for technical and pedagogical assistance.
    - QEMU Emulated board or physical board connected to the online PC (depending on the course).
    - Some Labs may be completed between sessions and are checked by the trainer on the next session.
  - For face-to-face trainings:
    - One PC (Linux ou Windows) for the practical activities with, if appropriate, a target board.
    - One PC for two trainees when there are more than 6 trainees.
  - For onsite trainings:
    - An installation and test manual is provided to allow preinstallation of the needed software.
    - The trainer come with target boards if needed during the practical activities (and bring them back at the end of the course).
- Downloadable preconfigured virtual machine for post-course practical activities
- At the start of each session the trainer will interact with the trainees to ensure the course fits their expectations and correct if needed

Prerequisites

- Knowledge of a programming language like C or C + +

Target Audience

- Any embedded systems engineer or technician with the above prerequisites.

Evaluation modalities

- The prerequisites indicated above are assessed before the training by the technical supervision of the traineein his company, or by the trainee himself in the exceptional case of an individual trainee.
- Trainee progress is assessed in two different ways, depending on the course:
  - For courses lending themselves to practical exercises, the results of the exercises are checked by the trainer while, if necessary, helping trainees to carry them out by providing additional details.
  - Quizzes are offered at the end of sections that do not include practical exercises to verifythat the trainees have assimilated the points presented
- At the end of the training, each trainee receives a certificate attesting that they have successfully completed the course.
  - In the event of a problem, discovered during the course, due to a lack of prerequisites by the trainee a different or additional training is offered to them, generally to reinforce their prerequisites,in agreement with their company manager if applicable.

Plan

First Day

Introduction

- History of Java
- Features of Java
  - Portability
  - Security
  - Robustness
  - Simplicity
  - Multithreading
- The JDK (Java Development Kit)
- The virtual machine
- The basics of JAVA
  - Data types
  - Operators
  - Flow control

Exercise: *Write the "Hello World" program in Java*

Object Programming in Java

- Object-oriented programming
  - Encapsulation
  - Inheritance
  - Polymorphism
  - Interfaces

Exercise: *Write a producer-consumer program in java (plant)*

- The nested classes and interfaces
  - Internal Classes
  - Anonymous Classes
- Typecasts and instanceof operator
- Packages
  - definition
  - import
  - search order

Exercise: *Rewrite the plant using anonymous classes*

Advanced aspects

- Generics in Java
  - Generic Classes (parameterized)

    - Generic Methods

Exercise: *Configuring the plant with generic types*

- Java exceptions
    - Presentation of exceptions and their mechanism
    - Capture and propagation of exceptions
    - Exception classes
    - Business exceptions

Exercise: *Controlling the plant with exceptions*

Second Day

Multitask programming in Java

- What is a thread
- The Java threading API
- Inter-thread synchronization
- Thread scheduling
- Asynchronous communication between threads

Exercise: *Create two plants working in parallel, the second consuming the products of the first*

Java utility classes

- Manipulating strings
    - The String class
    - The StringBuffer class
- Input/Output
    - The java.io package
    - Standard I/O read and write
    - Reading and writing text files

Exercise: *Write a program that reads a text file and print one word per line*

- Mathematical computations:
    - The java.lang.Math class
- Date management
    - The Calendar class
- Internationalization
    - The Locale class
- Environment access
    - The System class
    - The Runtime class

Exercise: *Modify the program to sort the result (using the "sort" command)*

Third Day

Data management in Java

- Collections
    - The collection types and interfaces
    - The collection abstract classes
    - The implementation classes
- The Iterator interface
- Comparing and sorting objects
- Rational use of collections

Exercise: *Rewrite the previous program to count the number of occurrences of each word and display the 10 most frequent*

Applets

- What is an Applet
- The Applet-specific APIs

- Declaring an Applet in HTML
- Applets and security
  - Signature and certificate
  - Generating a signed Applet
- Communicating between Applets
- Communicating with the browser

The Java Native Interface (JNI)

- Interfacing Java code with C/C++ code
- Overview of the Java Native Interface
- Calling a native method or function
- Naming conventions for called functions
- Passing Java objects to C/C++ code
- Accessing Java objects from C/C++ code
- SWIG (Simplified Wrapper and Interface Generator)
  - Interfacing Java code to existing C/C++ code.

Fourth Day

Packages, Interfaces and "jar" files

- Creating a Package
- Creating an Interface
- Creating a Jar file

Security in Java

- Security of the Java 2 platform
  - The Class Loader
  - Security Domains
  - The Access Controller
- The Security Manager
  - Security rules files
  - Permissions
  - The FilePermission class
- Cryptography
  - Digital signatures
  - Certificates

Optimization

- Just in Time (JIT) compiler
- Ahead of Time (AoT) static compiler
- Choosing the compiling mode
- Some rules to write efficient code
- Monitoring tools

Renseignements pratiques

Inquiry : 4 days