

# Java Fundamentals Course

(version 1.0)

## I. Overview

The course objective is to give the students a stable basis of knowledge and skills for working as a software engineer for the Java platform. The course covers the fundamental set of knowledge and skills that any Java developer should have: Java language, object-oriented programming, standard Java APIs, Swing GUI applications, working with Threads, Images. The course is based on Java 6, Eclipse 3.4

## II. Requirements

Elementary computer skills: required.

Computer English: initial level.

Preliminary programming skills: required (Introduction to Programming with Java).

## III. Course Program

### 1. Course Overview (lectures: 2, exercises: 0, homework: 0)

- Course scope, curriculum, homework, lectures and exercises
- Introduction to the course environment: the Subversion repository and TortoiseSVN

### 2. Java Platform Architecture (lectures: 2, exercises: 1, homework: 3)

- The Java Virtual Machine (JVM)
- Creating, compiling and running Java applications
- The structure of JAR archives
- Classpath and class loader

### 3. Java Programming Language (lectures: 2, exercises: 2, homework: 6)

- Java programs structure
- Identifiers; Keywords
- Primitive data types; Enumerations
- Variables; Assignments; Operators;
- Expressions; Statements; Control structures (if, for, while, do ... while)
- Console input/output

- Arrays – one-dimensional and multidimensional arrays
- Using the Java API Documentation

#### **4. Object-Oriented Programming Concepts (lectures: 2, exercises: 0, homework: 2)**

- Object-oriented programming and design fundamentals
- Classes, objects and interfaces
- Inheritance, abstraction, encapsulation and polymorphism

#### **5. Classes and Objects (lectures: 2, exercises: 2, homework: 2)**

- Creating and using objects; Value types and references
- Classes, constructors, methods, fields, constants, access modifiers
- Packages
- Inner classes
- Static methods and static constructors

#### **6. Inheritance and Polymorphism (lectures: 1, exercises: 2, homework: 3)**

- Inheritance and polymorphism
- Abstract classes and interfaces

#### **7. Exceptions Handling (lectures: 1, exercises: 1, homework: 2)**

- Error handling and exceptions
- Catching and throwing exceptions
- Defining and using own exception classes

#### **8. High-Quality Code Construction (lectures: 2, exercises: 0, homework: 0)**

- What is high-quality programming code?
- High-quality software design (cohesion, coupling, complexity, reusability, fan-in, fan-out, etc.)
- High-quality methods (cohesion, coupling, naming, parameters, length, etc.)
- Defensive programming and exceptions
- Best practices using variables (naming, scope, span, lifetime, etc.)
- Naming variables, methods, classes and other identifiers
- Best practices using conditionals and loops
- Self-documenting code and effective comments

## **9. Working with Strings (lectures: 2, exercises: 2, homework: 12)**

- The Unicode standard
- Strings and string processing: creating, concatenating, extracting substrings, searching, comparing, and splitting
- Parsing and formatting numbers
- Building and modifying strings: StringBuilder and StringBuffer classes

## **10. Dates, Locales and Internationalization (lectures: 1, exercises: 2, homework: 2)**

- Dates and calendars
- Locales, string formatting, formatting numbers and dates
- Parsing numbers and dates
- Character sets and conversions
- Localization with ResourceBundle
- Date/Time API for Java 7 – Joda Time

## **11. Working with Objects (lectures: 2, exercises: 2, homework: 3)**

- java.lang.Object, equality, hash codes, toString()
- Cloning objects
- Comparing objects
- Type conversions and casting
- Value types vs. Objects; Autoboxing/unboxing

## **12. Algorithms and Complexity (lectures: 2, exercises: 0, homework: 2)**

- Algorithms complexity
- Asymptotic notation
- Analyzing and computing complexity

## **13. Arrays Manipulation (lectures: 1, exercises: 0, homework: 1)**

- Standard algorithms on arrays: sorting, binary search, printing

## **14. Generics (lectures: 1, exercises: 1, homework: 3)**

- Generic methods
- Generic classes and interfaces

## **15. Data Structures (lectures: 2, exercises: 2, homework: 8)**

- Lists and representation
- Queues and representation
- Stacks and representation
- Trees, binary trees and balanced search trees
- Sets and representation
- Maps, hash-tables and representation

## **16. Java Collections Framework (lectures: 2, exercises: 2, homework: 8)**

- List Collection Classes – ArrayList<T>, Vector<T>
- Map Classes – Hashtable<K, T>, HashMap<K, T>, TreeMap<K,T>
- Set Classes – HashSet<T>, TreeSet<T>
- Java Collections Framework
- Collection Interfaces – Collection<T>, List<T>, Set<T>, Map<K, T>, Iterable<T>, Iterator<T>
- Collection Wrappers and Algorithms

## **17. Graphs and Traversal Algorithms (lectures: 1, exercises: 2, homework: 4)**

- Graphs – basic concepts and representation
- Depth First Search (DFS) algorithm
- Breath First Search (BFS) algorithm
- Traversing the file system (with DFS and BFS algorithms)

## **18. Regular Expressions (lectures: 1, exercises: 2, homework: 4)**

- The regular expressions language – literals and meta-characters, character classes, quantifiers, grouping characters, etc.
- Regular expressions in Java
- Searching, extracting, validating, splitting and replacing text by regular expressions

## **19. Input/Output – Streams and Files (lectures: 2, exercises: 3, homework: 8)**

- Streams
- Text streams (readers and writers) and text files
- Binary streams and binary files

## **20. Input/Output – Advanced Topics (lectures: 1, exercises: 2, homework: 8)**

- Console I/O, scanners and formatters
- Files and directories
- Accessing CLASSPATH resources
- Advanced streams and special I/O classes

## **21. Annotations (lectures: 1, exercises: 0, homework: 2)**

- Using annotations, annotation parameters, annotation targets

## **22. Threads and Multithreading (lectures: 2, exercises: 2, homework: 4)**

- Multithreading (Thread and Runnable)
- Timer and TimerTask
- Thread-local variables (ThreadLocal<T>)
- Thread pools; Executors
- Nanotime API

## **23. Threads Synchronization (lectures: 2, exercises: 2, homework: 4)**

- Race conditions
- Basic synchronization techniques: synchronized methods and blocks
- Monitors
- Atomic variables
- Using wait() and notify()
- Avoiding dead locks
- Classical synchronization problems: producer / consumer problem
- Concurrent collections

## **24. Reflection (lectures: 1, exercises: 1, homework: 2)**

- Loading classes
- Exploring metadata – classes, methods, class members
- Dynamically instantiating classes and invoking methods

## **25. Serialization (lectures: 1, exercises: 1, homework: 1)**

- Automatic and custom serialization / deserialization

## **26. Creating GUI with Swing (lectures: 2, exercises: 3, homework: 6)**

- AWT/Swing programming model, basic classes
- JavaBeans component model – components, properties, events
- Frames and dialogs (JFrame, JDialog)
- Adding components to the frames
- Handling events (events, event sources and listeners)
- Basic components (JLabel, JTextField, JButton, JPanel, JScrollPane)
- Introduction to Jigloo GUI editor

## **27. Advanced Swing Programming (lectures: 3, exercises: 3, homework: 10)**

- Layout managers
- Advanced components (menus, status bars, toolbars, tables, trees)
- Using Threads in Swing

## **28. Basic Networking Concepts (lectures: 1, exercises: 0, homework: 2)**

- OSI seven-layer model
- TCP/IP networking model: protocols, services, IP address, network interface, DNS
- TCP and UDP sockets

## **29. Network Programming in Java (lectures: 2, exercises: 2, homework: 12)**

- Using TCP sockets: client and server sockets
- Creating multithreaded socket servers
- UDP sockets: sending and receiving packets
- Accessing Internet resources through URL
- E-mail API

## **30. Javadocs (lectures: 1, exercises: 0, homework: 1)**

- What is javadoc
- Structure of javadoc comment
- The most frequently used tags
- Generating javadoc in Eclipse

## **IV. Training Duration**

Lectures: 48 hours

National Academy for Software Development

Web-site: <http://www.nars.bg>

Page 6 of 7

Copyright (c) 2009 National Academy of Software Development, Ltd. All rights reserved. Unauthorized copying or re-distribution is strictly prohibited.

Exercises: 41 hours

Homework: 122 hours

Allocation: ~ 15 weeks, 2 times \* 3 hours at week