Core / Intermediate Java

Class Outline

Simon Roberts, January 2016

Getting started

- Create Hello World project
- Create a class: package name, main entry point
- Correspondence between class/package and file/directory
- Lower case package names
- System.out.println
- Build / Run from command line
- Everything is defined inside classes
- Reading from command line: Scanner
    - import statements, and IDE shortcut

Lab: Create and run a Hello World example

Variables and Types

- Primitive types: int, long
    - Literal forms: 0xddd, 0dd, 0b in J7, Underscore
    - L suffix
- Floating point: float, double, default literal is double
    - F suffix
- Initialize before use rule
- Java type safety, cast operations with primitives
- Arithmetic promotions
- Identifiers: letter, letters & numbers, ($, _)
- Naming conventions, importance of names
- boolean type: only boolean is boolean, true, false
- Overview of operators
    - Arithmetic: + - * / %
    - Comparison: < <= >= > == !=
    - Logical: && || !
    - Conditional: ? :

Lab: Zeller's Congruence 1, Temperature Converter

More Types

- (byte, short)
- char: unsigned, literal form uses single quotes, Unicode literals and escapes
- String:
    - Immutability
    - Overloaded plus operator, universal conversion

    - == vs equals: Invoking methods and meaning of equality (primitive vs reference types)
    - First look at Java documentation
    - Operations: charAt, length, subString, toUpperCase, toLowerCase, trim, toCharArray, toString
- StringBuilder
    - Operations: append, charAt, delete, deleteCharAt, replace, setCharAt, substring, toString
    - equals doesn't work
    - Construction from String
- Simple enum types (values only) as stand alone classes in their own source files
- Key enum operations: valueOf(String), values() and ordinal()

Lab: String Chewing, That's Mister To You! Guessing Game Setup

Control structures

- if/else: requirement for boolean
- switch/case/default: int, enum, String
- while & do/while
- for(; ;)
- break/continue: labeled

Lab: Zeller's Congruence 2, Las Vegas 1, In Range 1, In Range 2, Text Alignment 1, Text Alignment 2, Wake Up

Array types

- Initialized literal form BaseType [] bta = { a, b, c[,] };
- Creating an uninitialized array new BaseType[n]
- Non-initializing literal form: fn(new BaseType[] { ... })
- Subscript access, index range
- Array is put in the heap
- Dot-length field, non-resizable
- System.arraycopy
- Arrays of arrays: rectangular and non-rectangular forms,
- Enhanced for loop

Lab: Las Vegas 2, 3a, and 3b, Code Breaker, Guessing Game

Public Static Methods

- Arguments, types and order
- Return type, return keyword, void return
- "Real" name, overloading
- Variable argument list form

Lab: Calendar, Palindrome Checker

Introduction to Java Libraries

- Finding and using Java documentation
- Introduction to Interfaces
    - Describing what you can do
    - Generalization concept, CharSequence interface
    - Object as ultimate generalization
    - Simple generic interface
- Collections
    - List, add, clear, contains, get, indexOf, isEmpty, lastIndexOf, remove, set, size, sort(null) [Java 8]
    - Overview of hashing, need for hashcode + equals
    - Set, Map

Lab: Word List, Concordance

Structured types

- Simple class definition
- Public fields
- Instantiation with new
- Accessing fields with the dot operator

Lab: Birthdays 1

Exceptions

- try, catch, finally
- throw, instantiating an exception, defining an exception
- throws
- Checked vs unchecked exceptions

Lab: Birthdays 2

IO and Networking

- InputStream, OutputStream, Reader, Writer
- BufferedReader, PrintWriter
- InputStreamReader, OutputStreamWriter
- Socket, ServerSocket
  - close, flush
- Exception handling using try-with-resources

Lab: File Server 1, File Server 2

Creating Simple Classes

- Instance methods
- Encapsulation, making fields private
- Making methods public, protected, default, or private
- Simple initialization
- Default/non-default constructors, this()
- Role of exceptions in encapsulation
- Static factory methods

Lab: Birthdays 3

Further Class Features

- General instance methods
- toString() method
- equals() / hashcode() methods
- Default access
- Access controls applied to data and behavior

Lab: Birthdays 4

Implementing Generic Interfaces

- Comparable<> / compareTo() method
- Comparator<> / compare method, List.sort(Comparator)

Lab: Birthdays 5, Birthdays 6

Reusing Behavior 1

- Defining your own interface
- Collections of interface types
- Strategy pattern, interface granularity

Lab: Custom Awards

Reusing Behavior 2

- Inheritance
  - Combinatorial explosion, run-time inflexibility
  - Accessing parent class features, super.xxx()
- More initialization
  - super()
  - static and instance initialization blocks
- Abstract classes
- The protected accessibility modifier

Lab: Zoo Keeper