Programmieren II

Summary & External Libraries

Alexander Fraser

fraser@cl.uni-heidelberg.de

July 16, 2014

Klausur

- **23/07/2013, 14:15–15:45**, here
- Bring paper!

Outline

Outline

Exam

- Written exam: a little less than 90 minutes
- Everything we covered up to, but not including, last week
- Material covered in assignments

Introduction and Types I

Basic Java sessions

- Why Java? Java vs. other programming languages
- Structure of a simple Hello World program

Variables and Types

- Strong typing (type of a variable cannot change)
- Static typing (variable can only take values of the variable's type)
- Data types for whole numbers and floats
- Issue of rounding errors when casting
- Numeric literals

Introduction and Types II

Operators

- Arithmetic operators
- Increment/decrement operators
- Comparison operators
- Logic operators
- Type conversion

$\rightarrow$ variable declaration and assignment of values

Reference types

- Arrays
- Multidimensional Arrays
- Classes

Control structures

- if-else
- while
- for
- break and continue
- for-each
- switch
- do-while

Class initialization and OOP I

Initialization and Constructors

- Initializing a class
- Constructors vs. static initializers
- Variable assignment (nullpointer) vs. object instantiation
- Overloading methods
- Constructor chaining

Modifiers

- static
- final
- public, private, protected

OOP continued

OOP

- Inheritance
- Overriding methods
- `Object` methods

- Strings and string methods
- String concatenation vs. `StringBuilder`
- String formatting

Additional Java topics

- Input/Output
- Exceptions
- Collections
- Polymorphism
- Abstract classes and Interfaces
- Sorting Collections and Sorted Collections
- Generics
- Enums

Software Design

- OOP Principles
- Unit Testing
- Java-Docs and Jar Files

Not on the exam

- Apache Ant
- Swing
- Event Loop
- Multi-threading
- Inner classes
- External libraries (today)
- XML, etc. (tomorrow)

Schedule

- Today 16.07: Summary, useful external libraries
- Thursday 17.07: XML/HTML, other useful things (attendance optional)
- Wednesday 23.07: Klausur
- Thursday 24.07: No class

Proseminar: Parallel Programming Paradigms

- Formally, there is no Prog III in Heidelberg
- However, I strongly recommend you take Parallel Programming Paradigms
- Proseminar, instructor: Schigehiko Schamoni (ICL)
- Very likely to be offered in SS 2015
- See 2014 course web page: `http://www.cl.uni-heidelberg.de/courses/ss14/parallelProgramming`

Outline

Core vs. New Functionality

- Core functionality changes the way you program Java
- New functionality lets you do new stuff, for instance, parse XML
- Two commonly used core libraries: Apache Commons Lang and Google Guava
- In general: be aware of licensing issues when working at a company

Apache Commons

- URL: `http://commons.apache.org/`
- Lang3, but also many other packages, e.g., Math

Apache Commons Lang

- Standard Java libraries are missing important methods for core classes.
- Lang3 provides:
  - String manipulation methods, basic numerical methods
  - Object concurrency, creation, serialization and System properties
  - Enhancements to java.util.Date
  - Utilities for building methods, such as hashCode, toString and equals

Google Guava Java library

- Collections, caching (as in dynamic programming), concurrency libraries, string processing, I/O, etc.
- Project website: `https://code.google.com/p/guava-libraries/`
- User guide: `https://code.google.com/p/guava-libraries/wiki/GuavaExplained`
- API documentation: `http://docs.guava-libraries.googlecode.com/git-history/release/javadoc/index.html`
- Also google for recent powerpoint/pdf presentations (changes rapidly)

Non-Core External Libraries

- These allow us to extend the functionalities of Java
- Apache Commons is a good place to start looking
- (Logging/Math materials in this section from T. Bögel)

Logging

Logging in Java

- Recommendation: Java Logging API
- Alternative: 3^{rd} party APIs (e.g. Apache Logging)

Creating a Logger

```
import java.util.logging.Logger;
private final static Logger LOGGER = Logger.getLogger(MyClass.
    class.getName());
```

Log Levels

Severity of a message

- SEVERE (highest)
- WARNING
- INFO
- CONFIG
- FINE
- FINER
- FINEST (lowest)

- Setting Logger to `info` level: logs `severe`, `warning` and `info`
- `LOGGER.setLevel(Level.INFO);`

Handler

- Handlers process log messages
- Standard handlers and custom handlers

Standard handlers

- `ConsoleHandler`
- `FileHandler`

Log Levels INFO and higher will be automatically written to the console.

Formatter

- Each handler can be configured with a formatter

Standard formatters

- `SimpleFormatter`
- `XMLFormatter`
- Custom formatter by extending `Formatter` and overriding `format` method

Notes

Best practises

- Custom logger for **each** class
- Name of the logger: fully qualified class name of the class
- Hints and documentation: `http://docs.oracle.com/javase/6/docs/technotes/guides/logging/`

Example: Apache Commons Math

Math component

- Distributions: `BetaDistribution`, `BinomialDistribution` etc.
- Statistic tests: `ChiSquareTest`, `PearsonsCorrelation` etc.
- Utility classes: `FastMath` etc.

Example: Commons Math I

Summary statistics for a list of double values:

```
// Get a DescriptiveStatistics instance
DescriptiveStatistics stats = new DescriptiveStatistics();

// Add the data from the array
for( int i = 0; i < inputArray.length; i++) {
        stats.addValue(inputArray[i]);
}

// Compute some statistics
double mean = stats.getMean();
double std = stats.getStandardDeviation();
double median = stats.getPercentile(50);
```

Example: Commons Math II

Correlation between two double arrays

```
PearsonsCorrelation correlation = new PearsonsCorrelation().
    correlation(x, y);
correlation.getCorrelationStandardErrors();
correlation.getCorrelationPValues();
```

Two-sample t-test

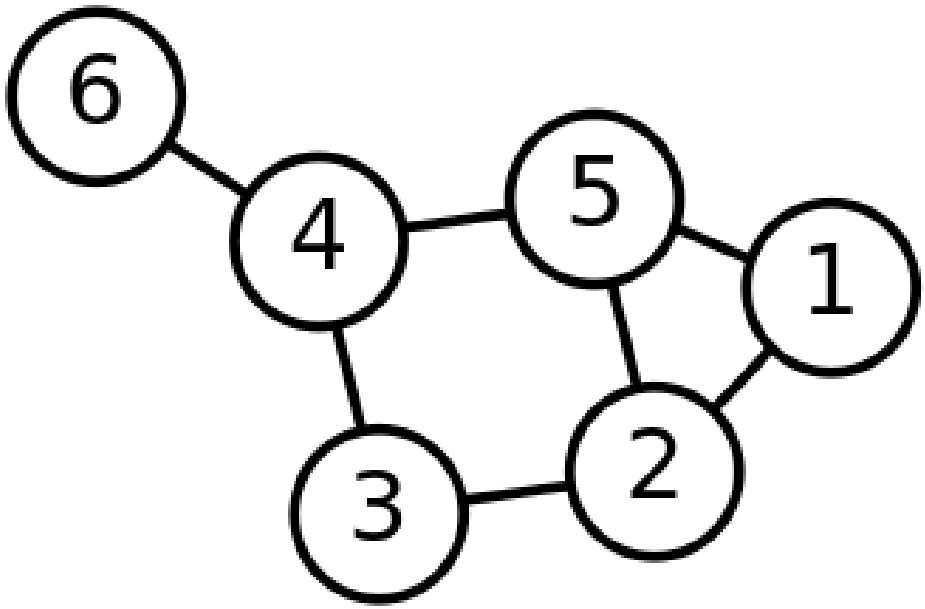
- T-Test for two samples (double arrays)

```
// compute t-statistics
TestUtils.pairedT(sample1, sample2);
// compute the p-value
TestUtils.pairedTTest(sample1, sample2);
```

Java graph library: JGraphT I

Graph

- Set of nodes and vertices between nodes



Java graph library: JGraphT II

JGraphT

- JGraphT: `http://jgrapht.org/`
- Graph library for graphs consisting of arbitrary objects
- Directed graphs, weighted graphs etc.
- Common useful graph algorithms already implemented (e.g. Dijkstra)
- Comprehensive API

Stanford CoreNLP Java API

Java CoreNLP

- POS tagger, named entity recognizer, parser, coreference resolution system, sentiment analysis
- Overview: `http://nlp.stanford.edu/software/corenlp.shtml`
- Getting started: `http://nlp.stanford.edu/software/corenlp.shtm`, click on "usage", then see the code snippet a couple of pages down
- See also Stack Overflow for questions on using CoreNLP and other Stanford tools

Apache OpenNLP

Apache OpenNLP

- Suite of useful NLP applications (as seen in assignment 8, also includes parser and coreference)
- Home page: `https://opennlp.apache.org/`
- API: `https://opennlp.apache.org/documentation/1.5.3/manual/opennlp.html`
- Tutorial: `http://www.programcreek.com/2012/05/opennlp-tutorial/`

Resources

- Google is your friend: particularly interesting are questions on Stack Overflow about how to do something
- Interesting list of commonly used external libraries (actively updated, not just from 2011) from X Wang:
  - http://www.programcreek.com/2011/08/the-most-widely-used-java-apis
  - Has example code in many cases