Techniques of Java Programming

Manuel Oriol, April 5th, 2006
Goal

• Teach Java to proficient programmers

• Teach basics and also advanced features

• Have people read research articles and specifications
Roadmap

- Java Basics
- Eclipse
- Threads and synchronization
- Class loading and reflection
- Java Virtual Machines
- Byte-Code and Just-In-Time compilation
- Java Middleware
- Java Components

and more...
Modus Operandi

- Assistants: Dr. Lisa Ling, Till G. Bay, Andreas Leitner
- Lectures:
  - 2 hours lectures on Thursday
  - 1 hour exercise on Wednesday
- Exercises will be corrected in the exercise lectures, not graded
- Written exam: 60% of the grade
- Project: 40% of the grade
Documents

• Slides - Web

• Course Abstract - Printed...

• Articles/Reading material - Web links
Java

• O–O language
• multi-platform
• type-safe
• class-based
• imperative
The Big Picture

Source Code

.java

Compiler

.javadoc

Documentation Generator

.Byte Code, Program

Documentation

.html

Virtual Machine

.class

.java

javac

.java

javadoc

java
Environment Variables

- PATH
- CLASSPATH
Java Compiler

- javac *.java
- arguments are files with extension .java
- each class has a .class file
- options:
  - -g for debugging
  - -verbose
  - -cp for classpath
  - -d for output directory
Java Virtual Machine (JVM)

- `java MyMain`
- arguments is a class name containing main method
- each class loaded on-demand
- options:
  - `-verbose`
  - `-cp for classpath`
  - `-d for output directory`
Java Documentation

- javadoc *.java
- comments /** */ and //</p>
- options:
  - verbose
  - cp for classpath
  - d for output directory
Java Archive

• jar cvf classes.jar *.class
• first argument is the target is option f
• each class loaded on-demand
• options:
  c create the archive
  v verbose mode
  f first argument is target file name
  x extract archive
Java Debugger

- `jdb MyMain`

- commands during execution:
  - `stop at MyMain:25`
  - `stop in MyMain.myMethod`
  - `next`
  - `step`
  - `run`
  - `print`
Java Basics
Part 2 - Language
Manuel Oriol, April 5th, 2006
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
Packages

- Package Names: general.lessGeneral.precise
- Defines a directory infrastructure
- Fully Qualified Name
Primitive Types

• Primitive types: int, byte, char, long, short, boolean
• comparisons: <, >, ==, >=, <=, !=
• operators: +, -, *, /
• Equivalent classes: Integer, Byte, Character, Long, Short, Boolean
Reference Types

- Classes
- Generic classes
- Interfaces
public abstract class MyClass{

}
Interfaces

• Only signature of methods

• has to be implemented in classes

```java
public interface MyInterface{

...

}
```
Inheritance

- Single inheritance
- Implementation of interfaces to simulate multiple inheritance

```java
class MyClass extends Object
    implements MyInterface{
}
```
Arrays

- Type 
- Declared when using variables
Generic Classes

• Classes dependent on another class
• (Will come back to this)

public class MyClass < E >{
    
    }


Variables

- Local Variables
- Instance Variables
- Class Variables
Local Variable

• Declared in the code (no matter the location)

• Local to the current block ({})

• Must be initialized before using them

```java
int a;
int b=3;
MyClass c;
String s=”Hello World!”;
```
Instance Variables

- Declared the same way as local variables but outside any method
- final, transient, volatile
- Visibility:
  - public
  - protected
  - default
  - private

```java
public int a;
private int b=3;
MyClass c;
protected String s="Hello World!";
```
Class Variables

- As instance variables but with static
- shared by all instances
this

• refers to the current object