Computer Programming

Java Archives (JAR)

Reflection

Willy Picard
Department of Information Technology
The Poznan University of Economics
<picard@kti.ae.poznan.pl>

Agenda

- Lecture Goal(s)
- The CLASSPATH
- Java archives
- Reflection
- Examples
- Conclusion

Lecture Goal(s)

Lectures Overview

Java

- 8: Summarizing Example
- 9: Standard library
- 10: GUI – AWT
- 11: GUI – Swing
- 12: IO programming
- 13: Network programming
- 14: Java archives and Reflection
- 15: Conclusions

LECTURE GOALS

Today's Goal

To provide programming knowledge about archives and reflection in Java

The CLASSPATH

Running Programs

- The Eclipse way
- The JDK way
  - `java <fullyQualifiedClassName> <args>`
- Example
  - `java pl.kti.CompProg.BasketDemo`
  - `java pl.kti.CompProg.Calc 12 64`
- Beware!
  - Compilation: `myClass.java → myClass.class`
  - Execution: `java myClass`

The CLASSPATH

- Used by `java` to find classes
- List of class directories and archives
- Archives
  - Zip and Jar files
- Separators
  - ; (Windows) or : (Unices)
- Example
  - `set CLASSPATH=.;c:\Fridge;c:\fridge.jar`
  - `export CLASSPATH=.:/fridge:/fridge.jar`

Packages and Directories

- `set CLASSPATH=.;c:\Fridge`
- Mapping
  - Package `pl.kti.CompProg`
  - Directory `pl\kti\CompProg`

  - Class `pl.kti.CompProg.Fridge`
  - File `pl\kti\CompProg\Fridge.class`

Java Archives

Zip and Jar Files

- Zip
  - By Phil Katz (PKWARE)
  - `.zip` extension
  - Compressed bundle of files and directories
- Jar
  - By Sun Microsystems
  - `.jar` extension
  - Zip files with an optional META-INF directory

The Jar Manifest

- The MANIFEST.MF file
- In the META-INF directory
- Manifest structure

```
Manifest-Version: 1.0
Created-By: 1.2 (Sun Microsystems Inc.)


Main-Class: pl.kti.CompProg.Fridge
```

The Jar Tool

- Creating a jar file
  - `jar cf myFile.jar Cat.class animals/`
- Creating a jar file with a manifest
  - `jar cmf myManifestFile myFile.jar *.class`
- Extracting a jar file
  - `jar xf myFile.jar`
- Updating a jar file
  - `jar uf myFile.jar newVersion.class`

Reflection

Reflection Overview

- Manipulating
  - Interfaces
  - Classes
  - Constructors
  - Methods
  - Fields
- The Reflection API
  - A set of objects representing the above concepts

The java.lang.Class

- Retrieving a `java.lang.Class` from an Object
  - `IAnimal animal = new Cat();`
  - `Class c = animal.getClass();`
- Get information about a `java.lang.Class`
  - `Class parent = c.getSuperclass();`
  - `String className = c.getName();`
  - `boolean isInterface = c.isInterface();`
- Creating a `java.lang.Class` object
  - `Class c = new Class(“pl.Cat”);`

Creating Objects

- Empty constructor
  - `Object o = c.newInstance();`
- Get information about constructors
  - `Constructor[] constrs = c.getConstructors();`
  - `Class[] paramTypes = new Class[]{String.class};`
  - `Constructor constr= c.getConstructor(paramTypes);`
- Creating an object
  - `Object[] params = new Object[]{“Felix”};`
  - `Object catFelix = constr.newInstance(params);`

Calling Methods

- Get information about methods
  - `Method[] methods = c.getMethods();`
  - `Class[] paramTypes = new Class[]{String.class};`
  - `Method method= c.getMethod(“eat”, paramTypes);`
- Calling a method
  - `Object[] params = new Object[]{“Felix”};`
  - `Object myResult = method.invoke(catFelix, params);`

Examples

Reflection Example

Dynamic Animals

EXAMPLES

Conclusion

Conclusion

- CLASSPATH
  - Often a source of trouble
- Java Archives
  - Make your program one click away! :-)
- Reflection
  - Advanced concept
  - Highly dynamic programs

See you next week