Praktikum aus Softwareentwicklung 2

WS 2006/2007, LVA 365.009

Abgabetermin: Mi. 23. 10. 2006, 17:15 **Erreichbare Punkte: 24**

Übung 1 – Grundlegende Java-Bibliotheken

Aufgabe 1: Filemanager 14 Punkte

Implementieren Sie eine Filemanager Klasse die folgendes Interface implementiert:

```
public interface FileManager {
  public void copy(String from, String to);
  public void copy(String from, String to, String charSet);
  public void move(String from, String to);
  public void convert(String file, Charset charSet);
  public void convert(String file, String charSet);
  public void setListModeOnlyFiles(boolean onlyFilesMode);
  public void setListModeDate(boolean dateMode);
  public void setListModeSize(boolean sizeMode);
  public void setListModeDescSort(boolean descSort);
  public void setListExtension(String ext);
  public String listDir(String dir);
  public void toUnix(String from, String to);
}
```

Die beiden Methoden **`copy`** kopieren eine Datei, die Methode **`move`** verschiebt eine Datei und die Methode **`convert`** konvertiert eine Datei von einem Character Set zu einem anderen.
Die Methode **`listDir`** zeigt den Inhalt eines Verzeichnisses an. Ist der Name des Verzeichnisses **`null`**, wird das aktuelle Verzeichnis genommen. Das Aussehen des Listings beziehungsweise die Anzahl der Spalten werden durch die 4 **`setListMode*`** Methoden beeinflusst. Standardmäßig werden nur die Dateinamen angezeigt, durch Setzen des **`onlyFilesMode`** werden jedoch nur mehr Dateien, aber keine Verzeichnisse angezeigt. Die Anzeige des Modifikationsdatums kann mit dem Schalter **`dateMode`** ein und ausgeschalten werden. Die Sortierung wird mit der Methode **`setListModeDescSort`** gesteuert und mit **`setListModeSize`** kann die Anzeige der Dateigröße zugeschaltet werden.
Sollen nur bestimmte Dateien gelistet werden, wird mit der Methode **`setListExtensio`**n eine Dateiendung gesetzt, welche das Listing auf Dateien dieses Types reduziert. Verzeichnisse sollen dennoch angezeigt werden, außer der Listing Modus ist auf die Anzeige von Dateien reduziert.
Achten sie auch auf eine schöne Spaltengestaltung der Ausgabe und kennzeichnen Sie Verzeichnisse extra.
Die Methode **`toUnix`** ersetzt bei einer Textdatei (Nur bei Dateiextension *.txt) alle vorkommenden Zeilenumbrüche von Windows (**`CarriageReturn`** und **`NewLine`**) in einen Unix Zeichenumbruch (**`NewLine Zeichen`**). Verwenden sie dazu einen FilterStream.
Unter der folgenden URL finden Sie Hinweise dazu, wie man einen Filter Stream schreibt: http://java.sun.com/docs/books/tutorial/essential/io/writingFiltered.html

Um die Klasse zu testen, schreiben sie ein Testprogramm, welche alle Methoden mit den kritischen Testfällen testet.

Beispiel für den Aufruf von listDir:

```
FileManagerImp fmi = new FileManagerImp();
fmi.setListModeDate(true);
fmi.setListModeSize(true);
fmi.setListModeDescSort(false);
fmi.listDir("C:\\temp");
```

Rückgabewert:

```
12.01.2004 08:43        6.770 MappingSQLs.txt
15.03.2004 07:53 Dir          praktikum
04.02.2004 10:50              0 wrapper.txt
```

Aufgabe 2: Java - Kommandointerpreter 10 Punkte

Implementieren Sie einen einfachen Kommandointerpreter in Java, mit dem Java - Klassen geladen werden und statische Methoden ausgeführt werden können. Der

andere Java-Programme vom aktuellen Verzeichnis aus gestartet werden können. Implementieren Sie zumindest drei vordefinierte Kommandos, `cd`, `help` und `quit`. Mit `cd` wird das aktuelle Verzeichnis gewechselt (kann dann z.B. mit `List` aus Aufgabe 2 angezeigt werden). Mit `help` wird eine Liste der möglichen Kommandos angezeigt, d.h. mindestens "`cd`" und "`help`" bzw. die Namen aller Java-Klassen im aktuellen Verzeichnis, die eine statische Methode `main` enthalten. Falls eine Klasse eine statische Methode `showDescription` implementiert, dann kann diese verwendet werden, um eine Kurzbeschreibung des Programms zu erhalten. Mit `quit` wird der Interpreter beendet.

Verwenden Sie das **Reflection API** `java.lang.reflect`, um dynamisch nach vorhandenen Klassen zu suchen! Ausgehend von der Eingabe des Benutzers (z.B. `MyCompressor`, `List` etc.) soll die entsprechende Klasse gesucht und dann die statische Methode `main` mit den angegebenen Parametern ausgeführt werden.

Hinweis: Um die Benutzereingabe zu parsen, d.h. um den Kommandonamen und die Parameter herauszufiltern, kann die Klasse `java.util.StringTokenizer` verwendet werden.

Bsp:

```
DOSprompt> cd c:\Java\beispiele\dosShell\
New directory: c:\Java\beispiele\dosShell\
DOSprompt> List
Directory C:\Java\beispiele\dosShell\
  CopyCommand.class    Fri Mar 17 14:59:04 CET 2000
  CopyCommand.java     Fri Mar 17 14:09:30 CET 2000
  DeviceNotReadyExcept Fri Mar 17 14:59:04 CET 2000
  DeviceNotReadyExcept Fri Mar 17 14:49:16 CET 2000
  DeviceNotReady.class Wed Oct 02 16:31:22 CEST 1999  keine main Methode
  DOS                  Wed Oct 02 15:41:54 CEST 1999
  DOS.class            Fri Mar 17 14:59:04 CET 2000
  DOS.java             Fri Mar 17 14:59:02 CET 2000
  List.class           Fri Mar 17 14:59:04 CET 2000
  List.java            Fri Mar 17 14:51:58 CET 2000
  MyCompressor.class   Fri Mar 17 18:59:04 CET 2000
  MyCompressor.java    Fri Mar 17 18:09:30 CET 2000
```

```
DOSprompt> Help
Available Options:
cd ... Change current directory          Fixer Text
help ... Show available options          Fixer Text
CopyCommand                              keine Methode showDescription()
List ... List Directory                  Beschreibung aus showDescription()
MyCompressor ... Compress/Decompress a File
quit ... Quit DOS
DOSprompt> CopyCommand /tmp/hugo hugo2
Error: File Not Found: \tmp\hugo (Das System kann die angegebene Datei
nicht finden)
DOSprompt> Quit
Sie koennen den Computer jetzt ausschalten ;)
```

Abgaberichtlinien (gelten auch für zukünftige Übungen)

Verwenden Sie eine IDE ihrer Wahl bzw. rein die Kommandozeile, jedoch mit JDK 1.4.x. Achten Sie jedoch darauf, dass ihre Programme auch rein von der Kommandozeile aus kompilierbar sein müssen, d.h. für die Abgabe erstellen Sie eine ZIP-Datei ausgehend vom Root-Verzeichnis ihrer Sourcen.

Bsp: Ihre Sourcen liegen im Verzeichnis `c:\java\uebung1` (z.B. Aufgabe 1 in Unterverzeichnis `aufgabe1` usw.) Erstellen Sie daher ein ZIP-File *inklusive* dem Verzeichnis `uebung1`.

Falls Sie zusätzliche Dokumente benötigen (um etwa ihre Annahmen, die Sie bei der Ausarbeitung der Aufgaben getroffen haben, zu dokumentieren), dann verwenden Sie dafür *reinen Text* (ASCII).

Senden Sie alle Dokumente (ZIP-File mit den Sourcen + zusätzliche Dokumente) rechtzeitig vor Abgabetermin an die Email-Adresse:

java-tutor@ifs.uni-linz.ac.at

Im Betreff beginnen Sie bitte immer mit `[<MATR.NR>_PRSE2_UE<Nr.>]`, d.h. für die Übung 1 schreiben Sie `[PRSE2_UE1]`, für die Übung 2 [PRSE2_UE2] usw.

Beachten Sie bitte die Termine für die Abgabe !!!

Ihren aktuellen Punktestand können Sie auch jederzeit unter der folgenden URL einsehen:

http://www.bioinf.jku.at/teaching/ws2006/pr-swe2/

Programmierrichtlinien (gelten auch für zukünftige Übungen)

Achten Sie auf Einrückungen im Code und einen sauberen Programmierstil, d.h. verwenden Sie sprechende Variablennamen, versehen Sie jede Klasse und Methode mit einem Kommentar-Kopf ihrer Wahl und kommentieren Sie Programmstücke, die für das Verständnis ihrer Lösung notwendig sind.