

IPD
Universität Karlsruhe
AVG, 2. OG
Adenauerring 20a
76131 Karlsruhe

Prof. Dr. Gerhard Goos Zi. 201 Tel. 0721/608-4760
ggoos@ipd.info.uni-karlsruhe.de

Mamdouh Abu-Sakran Zi. 232 Tel. 0721/608-8350
msakran@ipd.info.uni-karlsruhe.de

Übungsblatt: 4

Ausgabe: 2.6. 2004

Besprechung: 9.6. 2004

Aufgabe: 4.1 (Reflexion)

Schreiben Sie ein Programm, das Klassennamen von der Kommandozeile einliest und die Schnittstellen der Klassen in Java-Syntax ausgibt (leere Methodenrumpfe sind zulässig). Wenden Sie das Programm auf sich selbst an.

Hinweise:

- Laden Sie Klassen mit `java.lang.Class.forName()`.
- Die Schnittstelle umfaßt folgende Informationen:
 - Interface oder Klasse?
 - Modifikatoren (`abstract`, ...)
 - Konstruktoren (siehe `java.lang.reflect.Constructor`)
 - Methoden (siehe `java.lang.reflect.Method`)
 - Felder (siehe `java.lang.reflect.Field`)

Aufgabe: 4.2 (Dynamische Erzeugung)

Schreiben Sie ein Programm, das einen Klassennamen sowie eine Liste von Argumenten von der Kommandozeile einliest und ein Objekt der gegebenen Klasse erzeugt, das mit den angegebenen Argumenten initialisiert wird.

Hinweise:

- Behandeln Sie Argumente zunächst als Zeichenketten.
- Eine `java.lang.Class` kann ihre Konstruktoren aufzählen. Wählen Sie einen beliebigen Konstruktor mit passender Parameterzahl.
- Ermitteln Sie die Parametertypen. Rufen Sie entsprechende Konstruktoren auf, die einen String als Parameter nehmen, um typisierte Argumentobjekte zu erzeugen.
- Dynamische Konstruktoraufrufe sind über `java.lang.reflect.Constructor.newInstance()` möglich.

Aufgabe: 4.3 (Dynamischer Aufruf)

Erweitern Sie die Lösung der vorangegangenen Aufgabe um das Einlesen eines Methodennamens und einer Liste von Argumenten sowie den dynamischen Aufruf der entsprechenden Methode auf dem erzeugten Objekt. Rufen Sie mit Ihrem Programm die Lösung von Aufgabe 1 auf, um die Schnittstellen der aktuellen Lösung auszugeben.

Hinweise:

- Eine `java.lang.Class` kann ihre Methoden aufzählen. Wählen Sie eine beliebige Methode passenden Namens und passender Parameterzahl.
- Behandeln Sie Methodenparameter analog zu Konstruktorparametern.
- Dynamische Methodenaufrufe sind über `java.lang.reflect.Method.invoke()` möglich.