Programmieren 1 – Schmiedecke

Laborexperiment X1 - Java-Kaffee ohne Milch und Zucker

Lernziele: Java-Entwicklung mit Konsolwerkzeugen und das Erstellen eines Laborberichts erlernen.

Java-Entwicklung mit Konsolwerkzeugen ist wie physikalische Berechnungen mit einem Rechenschieber: Es funktioniert immer, auch wenn keine Batterien zur Verfügung stehen, aber es ist **sehr umständlich** und erfordert große Disziplin. Lernen Sie heute die wichtigsten Konsolwerkzeuge kennen.

In diesem Kurs werden wir sonst eine **professionelle Entwicklungsumgebung** benutzen nur hin und wieder mit den Konsolwerkzeugen arbeiten, damit Sie damit umgehen können - und so Java programmieren, auch wenn keine komfortable Entwicklungsumgebung zur Verfügung steht. Es ist eine gute Idee, wenn Sie eigeninitiativ häufiger fertige Übungslösungen von der Konsole aus kompilieren und starten, um "auf dem Rechenschieber fit" zu bleiben.

Ein wichtiges Lernziel in diesem Kurs ist das **Erstellen nützlicher technischer Dokumentationen**. Sie werden zu jeder Wochenübung ein schriftliches Laborprotokoll anfertigen, das bewertet wird. Das sind nicht einfach Aufsätze, sondern technische Dokumente, die unter anderem automatisch generierte Programmdokumentationen, Fehlerdokumentationen und Bildschirmabzüge enthalten. Sie werden merken, wie ihnen die Erstellung der Berichte immer schneller von der Hand geht - und wie Sie auf Ihre eigenen Berichte zurückgreifen, wenn Sie irgendwo "steckenbleiben".

Bewertung: Laborprotokoll der Gruppe (in Moodle hochladen), Abnahme (einzeln) am Rechner

Wichtig: Bei dieser Übung wird es Riesenunterschiede hinsichtlich der Vorkenntnisse geben. Wenn Ihnen alles viel zu einfach ist, erkunden Sie weitere Java-Werkzeuge wie jar und javadoc. Schreiben Sie in Ihren Bericht, welche Teile Sie warum übersprungen haben. Aber vor allem: Helfen Sie Kommilitonen, die mit der Arbeitsumgebung kämpfen!

Vorbereitung

- Lesen Sie die untenstehende Arbeitsanleitung durch und machen Sie sich einen Arbeitsplan, indem Sie entweder durch Anstreichen oder als Stichwortliste die Arbeitsschritte planen. Wenn möglich, schätzen Sie den Zeitbedarf für jeden Schritt.
- Planen Sie auch, wie und wo Sie Ihre Daten sichern wollen.

Nachbereitung: Laborprotokoll

Ihr Protokoll soll immer folgendes enthalten:

- Ihren Namen und die Übungsnummer
- Die Übungsbezeichnung ("Java-Kaffee ohne Milch und Zucker")
- Die Zielsetzung der Laborarbeit (in wenigen eigenen Worten, gefolgt von einem Link auf diese Seite)
- Notizen die Teilarbeiten und freiwilligen Experimente im Labor (gern auch über die Java-Installation zu Hause) mit Hinweisen zu Schwierigkeiten, Fehlern oder neuen Erkenntnissen.
- Screenshots (unter Windows kopiert Alt+Druck das selektierte Fenster in die Zwischenablage, von wo Sie es in beliebige Dokumente einfügen können) zu wichtigen Ergebnissen oder Problemen. Dieser Teil ist zentral: Es soll ein für Sie selbst nützliches Dokument werden, in dem Sie selbst nachschlagen, wenn Sie sich nicht mehr genau erinnern (das passiert garantiert!).
- Quelltext-Auszüge – nur die unmittelbar betroffenen Zeilen -, auf die sich das Gesagte bezieht.
- Alle Quelltexte von Programmen, die Sie geschrieben oder geändert haben (außerhalb des eigentlichen Protokolls, d.h. als Link oder als Anhang).
- Alle Hilfen, die Sie in Anspruch genommen haben: Kommilitonen, Bücher, Internet-Ressourcen
- Eine Einschätzung, was Sie gelernt haben
- Den Zeitaufwand für das Labor und die Berichterstellung

Das Laborprotokoll sollen Sie grundsätzlich außerhalb der Übung fertigstellen. Allerdings ist es unbedingt erforderlich, während der Übung die Notizen und zugehörigen Screenshots zu sammeln. Mit ein bisschen Diziplin erstellen Sie sie so, dass sie sie nicht mehr nachbereiten müssen, so dass der Hauptteil des Protokolls dann bereits fertig ist.

Arbeitsanleitung:

Inhalt:

1. Kennenlernen der Arbeitsumgebung
2. Arbeiten mit der Kommandozeileneingabe
3. Java-Entwicklung von der Kommandozeile
4. Ein bisschen Spielen zur Belohnung

1. Kennenlernen der Arbeitsumgebung

Ihr Login gilt nur für das SWE-Labor, allerdings in beiden Laborräumen (D-E16a und D-E16b).

Sie erhalten mit Ihrem Login ein Festplattensegment auf dem Datenserver des Labors zugeteilt, Laufwerksbuchstabe "Z", das bis zum Ende des Semesters bestehen bleibt und für die Daten aus

dieser LV ausreicht. Auch wenn wir in diesem Labor noch keine Datenpannen hatten, ist es empfehlenswert, dass Sie Ihre **Übungsdaten auch noch an einer zweiten Stelle ablegen**, z.B. auf Ihrem HRZ-Bereich, oder auf einem USB-Stick oder inm eimner Dropbox. **Ein Datenverlust durch Nichtbeachtung geht zu Ihren Lasten.**

Wenn Sie auf Ihrem eigenen Notebook arbeiten wollen, sind Sie für die Installation und die Konfiguration der Arbeitsumgebeung selbst verantwortlich. Installieren Sie bitte das ***Java-8-SDK*** *, nicht das JREund* ***eclipse (Mars)*** *für Java-Entwickler. Sie sollten außerdem irgendeinen Texteditor installiert habern, ich empfehle* ***Notepad++*** *. Näheres zur Java-Installation finden Sie unter 3.*

Für Mac-User gibt es eine Installationsanleitung von Oracle:
https://docs.oracle.com/javase/8/docs/technotes/guides/install/mac_jdk.html#A1096855

Lernen Sie Ihre Arbeitsumgebung in einigen Schritten kennen. Probieren sie mehr aus, wenn Sie mögen - fragen Sie Ihren Dozenten oder Kommilitionen, wenn Sie nicht zurecht kommen. Wenn Ihnen alles vertraut ist, helfen Sie anderen!

1. Schauen sie sich an, welche Programme auf Ihrem Rechner installiert sind. Wir benötigen Notepad, Textpad, eclipse und das CommandPrompt. Sie können Verknüpfungen zu diesen Programmen erstellen (rechte Maustaste > Verküpfung erstellen) und diese direkt in das Startmenü schieben, dann können Sie sie schneller aufrufen. Solche Anpassungen der Arbeitsumgebung werden in Ihrem Profil gespeichert, sind also bei Ihrem nächsten Login noch vorhanden. (Achtung: Das Profil belegt Speicherplatz auf Ihrer persönlichen Festplattensektion).

2. Öffnen Sie die Dateiverwaltung (Aufruf des Windows Explorers durch Doppelklick auf das Icon) und finden Sie Ihren persönlichen Ordner als Laufwerk Z:. Öffnen sie ihn (er sollte leer sein) und erzeugen sie einen neuen Ordner "Uebung1" (rechte Maustaste > Neu > Ordner). Erzeugen Sie darn einen neuen Ordner "Test". Öffnen Sie den Ordner Test - Sie sehen oben in der Adresszeile den Pfad zu diesem Ordner: Z:\Uebung1\Test. (Wenn nicht, sagen Sie Bescheid, dann müssen wir wieder eine Einstellung vornehmen, die im Profil abgespeichert wird).

3. Öffnen Sie jetzt das Programm Notepad. Tippen Sie das folgende weltberühmte java-Programm ein:

```
public class Hello
{
	public static void main(String [] arguments)
	{
		System.out.println("Hello World!");
	}
}
```

4. Speichern Sie das Programm im Ordner Uebung1 unter dem Namen Hello.java . Schauen Sie in der Dateiverwaltung nach, ob sie es dort finden (falls nicht, drücken Sie F5 - dann wird das Fenster aktualisiert).

5. Sicher haben Sie auch Ihr Eliza-Programm irgendwo gespeichert. Speichern Sie es jetzt erneut im Ordner Uebung1 unter dam Namen Eliza.java (unter der Annahme, dass "Eliza" der Name Ihrer Klasse ist – sonst entsprechend).

2. Arbeiten mit der Kommandozeileneingabe ("DOS-Modus")

Mac-und Linux-Benutzer arbeiten nicht mit einer DOS-Shell, sondern mit einer Bash-Shell:
Die Kommandos sind fast gleich: dir heißt "ls" und type "cat".
Hier gibt es Hilfe zur Ausführung von Java-Programmen (falls nötig)
http://introcs.cs.princeton.edu/java/15inout/linux-cmd.html

1. Bisher haben Sie wahrscheinlich fensterorientiert gearbeitet - jetzt rufen wir die Betriebssystembefehle wie Öffnen einer Datei oder Starten eines Programms nicht per Mausklick auf ein Icon auf, sonderm indem wir Befehlstexte eintippen. Öffnen Sie dazu die Kommandozeileneingabe (CommandPrompt, Command Shell). Sie können dies fensterorientiert tun - Doppelklick auf Kommandoeingabe - oder bereits zeilenorientiert: Start > ausführen, "**cmd.exe**" eintippen. Probieren Sie beides aus. Probieren Sie auch, die Dateiverwaltung zeilenorientiert zu starten: Das Programm heißt **iexplore.exe**. (Die Endungen .exe und .bat bezeichnen vom Betriebssystem ausführbare Programme. Man kann diese Endungen weglassen). Tippen Sie jetzt auch "iexplore" in das schwarze Kommandozeilen-Fenster ein und starten Sie so die Dateiverwaltung. (Schließen Sie zwischendurch Programme, die Sie nicht benötigen...)

2. In der Kommandozeile arbeiten Sie immer von einem bestimmten Ordner der Dateiverwaltung aus; man bezeichnit ihn auch als Ihr **aktuelles Arbeitsverzeichnis** (Verzeichnis und Ordner sind Synonyme). Die wichtigsten Befehle sind **cd, dir** und **type**. Mit cd ("change directory") können Sie im Verzeichnisbaum navigieren, mit dir den Inhalt des aktuellen Arbeitsverzeichnisses ansehen, mit type den Inhalt einer Datei ansehen, sofern er aus lesbaren Zeichen besteht.

cd und dir haben als Parameter einen absoluten oder relativen Pfad. Ein absoluter Pfad beginnt(in Windows) mit der Bezeichnung des aktuellen Laufwerks, also z.B C:\Documents and Settings\All Users. Ein relativer Pfad beginnt beim aktuellen Arbeitsverzeichnis. also z.B: **cd Test**, wenn das Arbeitsverzeichnis Uebung1 ist. Ein Punkt "." steht für das aktuelle Verzeichnis, zwei Punkte für das unmittelbar darüber liegende. Mit **cd ..\..\..** können Sie also 3 Verzeichnisse "aufsteigen". Ein Laufwerkswechsel kann nicht im Zuge einer Navigation erfolgen, sondern muss explizit gemacht werden: "**X:**". dir und type nehmen als Parameter (auch) einen Dateinamen mit oder ohne Pfadangabe und mit oder ohne * als Platzhalter ("Wildcard"). **dir *.txt** listet alle .txt-Dateien im aktuellen Arbeitsverzeichnis auf, **type ..*.java** schreibt den Inhalt aller .java-Dateien im nächst höheren Verzeichnis auf den Bildschirm

Üben Sie selbständig den Umgang mit diesen DOS-Befehlen, z.B. indem Sie in das neue Test-Verzeichnis navigieren und von dort aus alle .java-Dateien im Uebung1-Verzeichnis herausschreiben lassen.

3. Alle Dateien mit der Endung .exe oder .bat sind Programme und wie DOS-Befehle ausführbar. Wenn sie im aktuellen Arbeitsverzeichnis stehen, können sie direkt mit ihrem Namen aufgerufen

werden, sonst mit Pfadangabe+Namen. Erstellen Sie im Testverzeichnis eine Datei Howdy.bat mit folgendem Inhalt:

```
echo How do you do?
```

Rufen Sie Howdy aus dem Testverzeichnis und, mit Pfadangabe, aus dem Uebung1-Verzeichnis auf.

4. Da die Angabe langer Pfade mühsam ist, kann man **dem Betriebssystem Pfade bekannt machen**, in denen es implizit nach ausführbaren Programmen suchen soll. Das geschieht durch Setzen der **Systemvariable PATH**. Sie kann eine beliebig lange Folge von Pfaden enthalten, die durch Semikolon getrennt sind. Die Reihenfolge bestimmt die Suchreihenfolge. Sehen Sie sich den gegenwärtigen Wert der Systemvariable PATH an, indem sie eintippen (Groß-Kleinschreibung ist hier beliebig, meistens aber wichtig...):

set path oder nur **path**

Einen neuen Wert setzt man durch Angabe des Pfades hinter einem Gleichheitszeichen:

set path=<neuer Pfad>

Aber Vorsicht, der alte Wert geht dabei verloren. deshalb schreibt man besser

set path=%PATH%;<neuer Pfad>

%Systemvariable% bezeichnet immer ihren gegenwärtigen Wert. Wir hängen also den neuen Pfad an den bisherigen Wert von PATH an.

(Achtung: war die Systemvariable bisher undefiniert, so wird die Zeichenkette "%Systemvariable%" anstelle des alten Wertes benutzt, was Unsinn ist!)

Hängen Sie den Pfad zum Test-Verzeichnis an Ihre PATH-Variable an. Navigieren Sie dann in irgendein anderes Verzeichnis und tippen Sie Howdy. Es sollte jetzt funktionieren. Spielen Sie ruhig ein bisschen herum.

5. Die mit set path gesetzte PATH-Variable gilt nur innerhalb des Kommandozeilen-Fensters, indem sie gesetzt wurde. Wird es geschlossen, hat sie beim nächsten Mal wieder ihren alten Wert. Es gibt die Möglichkeit, die Variablen so zu setzen, dass ihre Werte im Profil gespeichert werden, also das jeweilige Kommandozeilen-Fenster überdauern. Das geschieht durch das Menü **Systemsteuerung>System>Erweitert>Systemvariablen**. Dort können Sie neue sogenannte "lokale" Variablen definieren, die nur für Sie gelten. Probieren Sie es aus!

3. Java-Entwicklung von der Kommandozeile

Hier erfahren Sie Schritt für Schritt, was erforderlich ist, um eine funktionierende Java-Umgebung zu installieren. Im Labor sind die meisten Schritte schon getan worden, aber Sie sollten sie kennen - und möglichst auch zu Hause durchführen.

Installation von Java:

(Dieser Teil muss im Labor nicht gemacht werden – es ist eine Anleitung für zu Hause!)

*1. Laden Sie sich kostenlos das **Java-8-SDK** (SoftwareDevelopment Kit, s.o.)) herunter. Es enthält u.a. einen Compiler und einen Interpreter, die von der Kommandozeile aus aufrufbar sind. Sie benötigen neben dem SDK stets auch die Dokumentation: Wenn Sie eine schnelle Internet-Verbindung haben, genügt es, den Link auf die Dokumentation in der Favoritenliste zu speichern; sonst sollten Sie sich die Dokumentation herunterladen.*

2. Wenn Sie die Installationsdatei ausgeführt haben, gibt es eine neue Umgebungsvariable JAVA_HOME. Sie enthält den Pfad zum Java-Installationsverzeichnis. Damit finden Programme, die Java benutzen, die aktuelle Installation (auch der Browser). Für die Arbeit im Kommandozeilen-Fenster müssen Sie aber noch die PATH-Variable setzen. Sie finden die ausführbaren Dateien, wie den Java-Compiler javac.exe und den Java-Interpreter java.exe, im Verzeichnis bin unterhalb des Installationsverzeichnisses von Java. Fügen Sie also diesen Pfad Ihrer PATH-Variable hinzu (oder $JAVA_HOME$\bin). Der Java-Compiler und der Java-Interpreter können jetzt von jedem Verzeichnis aus aufgerufen werden.

Übersetzen und Ausführen von Java-Programmen

(Im Labor beginnen Sie bitte hier:)

3. Probieren die Installation aus, indem Sie java oder javac eintippen! (.exe kann man weglassen).

4. Der obige Test gab sicher eine lange Fehlermeldung. Verstehen Sie sie? Versuchen Sie immer, Fehlermeldungen zu lesen und so weit wie möglich zu verstehen...Beim Compiler-Aufruf wurde Ihnen mitgeteilt, dass der Befehl Parameter erwartet: Optionen, mit denen man die Kompilation beeinflussen kann (damit arbeiten wir später), und vor allem eine Quelldatei mit dem zu übersetzenden Programm. Um das vorhin eingetippte Programm Hello.java zu übersetzen, **wechseln Sie in das Uebung1-Verzeichnis** und tippen jetzt den Compiler-Aufruf mit Parameter ein:

javac Hello.java

Das sollte hoffentlich ohne Fehlermeldungen klappen, so dass wir das übersetzte Programm sofort ausführen können:

java Hello

5. Nun steht kein Java-Programm für sich allein, es benötigt mindestens die Klassen der Standardbibliotheken, um ausführbar zu sein (System und out und println sind in der Standardbibliothek java.lang definiert). Woher weiß der Compiler, wo diese liegen? Die Standardbibliotheken sind dem Compiler (und Interpreter) implizit über das Installationsverzeichnis bekannt gemacht (Umgebungsvariable JAVA_HOME).

6. Werden weitere Klassen benötigt, so sind ihre Pfade über eine weitere **Umgebungsvariable** bekannt zu machen: **CLASSPATH**. Diese Variable kann, genauso wie PATH, für die Lebensdauer eines Kommandozeilen-Fensters oder permanent gesetzt werden, oder auch nur für die Dauer einer Kompilation oder Ausführung durch die Option -cp. Um das auszuprobieren, verändern wir Hello jetzt. Dazu nutzen wir einen anderen Editor**:** **Textpad** oder **notepad++.** Speichern Sie die anfangs leere Datei sofort unter Hello2.java; die Endung .java ist eine wichtige Information für Textpad, denn

dann kann er Sie beim Erstellen des Java-Programms unterstützen. Tippen Sie einfach das folgende Programm ein und speichern Sie es. Was beobachten Sie?

```
public class Hello2 {
    public static void main (String[] arguments) {
         System.out.println(HelloText.gruss);
    }
}
```

Im Verzeichnis Test speichern Sie jetzt unter HelloText.java die folgende Datei:

```
public class HelloText {
   public static String gruss = "Hi Leute";
}
```

Hello2 greift auf die Klasse HelloText zurück, **die in einem anderen Verzeichnis** steht. Der Versuch, Hello2 zu kompilieren, scheitert, weil die Klasse HelloText nicht gefunden wird. Stünde sie auch im Uebung1-Verzeichnis, wäre das kein Problem (probieren Sie es aus!). So aber müssen wir die **CLASSPATH**-Variable auf das Test-Verzeichnis setzen, damit die Kompilation gelingt:

set CLASSPATH=%CLASSPATH%;<Pfad zu Test-Verzeichnis>

7. Ein guter Tipp ist, beim ersten Setzen der CLASSPATH-Variable immer das aktuelle Verzeichnis und das darüberliegende anzugeben: set CLASSPATH=.;.. (Warum?)

Ausführen von JAR-Dateien:

8. Beim Umgang mit den Komandozeilenwerkzeugen muss noch etliches mehr beachtet werden. Z.B. ist es sinnvoll, für die übersetzten Klassen ein anderes Verzeichnis zu benutzen als für die Java-Quellen (Warum?). Außerdem gibt es für realistische Java-Programme noch eine Strukturierung in sog. Pakete, die sich in der Verzeichnisstruktur wiederspiegeln muss. Aber da wir hier nur erste Schritte gehen wollen, brauchen wir nur noch eins:

9. Java bietet die Möglichkeit, mit **komprimierten Dateien, sog. Java-Archiven**, zu arbeiten. Sie heißen JAR-Dateien (wobei Jar auch "Deckelglas" bedeutet) - und können einfach mit Kompressions-Programmen wie WinZip oder WinRar bearbeitet werden (Tipp). Das SDK bietet das Programm jar.exe zur Erstellung von Jar-Dateien an. Eine solche Jar-Datei umfasst typischerweise alle für ein Programm erforderlichen Klassen und Zusatzdateien, wie Bilder etc., oder eine Klassenbibliothek, sowie eine Manifest-Datei, die den Inhalt beschreibt. Ein Programm, das in einer Jar-Datei gespeichert ist (und im Manifest als ausführbar beschrieben ist), kann direkt, d.h. ohne Entpacken, ausgeführt werden, indem man den Interpreter mit der Option -jar aufruft:

java -jar Quelle.jar

Alle Labor-Experimente liegen in Jar-Form vor, die "Demos" können direkt aus der jar-Datei heraus ausgeführt werden.

Benutzung von Bibliotheken:

10. Zur Benutzung Jar-Dateien als Bibliotheken muss im CLasspath der Pfad einschließlich der Jar-Datei angegeben werden:

set CLASSPATH=%CLASSPATH%;<Pfad>/Bibliothek.jar

Auf diese Weise kann z.B. die CS101-Bibliothek bekannt gemacht werden. (In den meisten Labor-Jar-Dateien ist die Bibliothek jedoch bereits komplett enthalten).

11. Laden Sie sich die **cs101-lib**-Bibliothek herunter, speichern Sie sie (irgendwo auf Z) und binden Sie sie über die CLASSPATH-Variable ein. Jetzt können Sie versuchen, Ihr Eliza-Programm zu übersetzen und auszuführen (es benötigt die Klasse Console aus cs101). Wenn es zu viele Fehler gibt, verschieben Sie die Arbeit mit Eliza auf das nächste Labor!

4. Ein bisschen Spielen zur Belohnung

1. Führen Sie das Programm javaeyes.jar* durch Doppelklick aus. Im Labor müssen Sie dazu evt. die Registrierung des 7zip-Programms für Jar-Dateien aufheben und java.exe dafür eintragen (Rechtsklick "Open with", java.exe auswählen, Häkchen für "immer mit diesem Programm öffnen" setzen).

2. Laden Sie sich die Datei javaeyes.jar herunter und führen Sie sie aus. (Was beobachten Sie?)

3. Entpacken Sie javaeyes.jar, setzen Sie den Classpath geeignet und führen Sie das Programm aus.

Genug für diese Übung - es gibt genug Material für Experimente. Probieren Sie aus, was Ihnen in den Sinn kommt oder wozu Sie Fragen haben.

Wenn Sie in wenigen Minuten "durchgekommen" sind, können Sie weitere Befehle erkunden (z.B. hier) und für Ihre Kommilitonen dokumentieren, die Java-Werkzeuge jar.exe und javadoc.exe erkunden - **oder vorzugsweise herumschauen, wer Ihre Hilfe gebrauchen könnte**.
