Programmieren 1 – Schmiedecke

Übung X2 – IDE-ales Java

PDF-Druckversion
Englische Version

Lernziele: Sie lernen in dieser Übung den Umgang mit der IDE Eclipse. Natürlich gibt es Hallo Welt und Eliza.. Uns geht es aber vor allem darum, dass Sie die Fingerfertigkeit bekommen, bestehende Programme in Eclipse-Projekte einzubringen und zu bearbeiten; denn die meisten Übungen in diesem Semester werden Arbeiten an bestehenden Programmen sein - gerade so wie im "richtigen Leben".

Vorbereitung

Lesen Sie den Aufgabentext. Finden Sie eine (deutsch- oder englischsprachige) Einführung in Eclipse, die Ihnen zusagt und merken sie sich den Link. Planen Sie ihr Protokoll.

Nachbereitung: Laborprotokoll

Legen Sie das Protokoll so an, dass es Ihr Eclipse-Handbuch wird. Sie dürfen es gern später fortschreiben, es muss nicht '"eingefroren" werden. Das Protokoll von Übung 3 passt gut dazu.

Inhalt:

Was ist eine IDE?

IDE steht für "Integrated Development Environment", zu Deutsch "integrierte Entwicklungsumgebung". Wir haben in der letzten Übung einen Vorgeschmack für das Arbeiten in einer nicht-integrierten Umgebung bekommen: Ein Programm wird mit einem Editor bearbeitet. Man musses gemäß bestimmter Konventionen im Dateisystem speichern, damit die anderen Werkzeuge es weiter bearbeiten können. Eine Reihe von weiteren Einstellungen sind erforderlich, um andere Werkzeuge die Datei weiterverabeiten zu lassen und Bibliotheken zugänglich zu machen.. (Möchte man zwischendurch an einer anderen Java-Aufgabe arbeiten, muss man typischerweise einzelne Einstellungen, insbesondere die Classpath-Variable, ändern.) Fehlermeldungen des Compilers oder Interpreters beziehen sich über die Zeilennummer auf den Quelltext - manchmal nummeriert der Editor anders als der Compiler. Bei größeren Projekten mit vielen Dateien und mehreren Bibliotheken wird das alles nicht nur kompliziert, sondern vor allem auch fehlerträchtig. Noch komplizierter wird die Angelegenheit, wenn ein Entwicklerteam gemeinsam an einem Projekt arbeitet...
Eine IDE integriert Editor, Compiler, Interpreter und etliche andere Entwicklungswerkzeuge zu einem Werkzeug. Der Programmierer muss nicht mehr zwischen verschiedenen Werkzeugen hin- und herwechseln und dabei die zugrunde liegende Verzeichnisstruktur und die Einstellung der Systemvariablen anpassen. Fehlermeldungen sind typischerweise direkt mit dem Quellcode verlinkt, sodass man per Mausklick an die Fehlerstelle gelangt. Weitere Annehmlichkeiten werden Sie im Laufe Ihres Studiums kennenlernen, wie Modellierungswerkzeuge, Versionsverwaltung oder Projektmanagement-Werkzeuge.
Die Arbeitseinheit einer IDE ist ein "Projekt". Zu einem Projekt gehören alle beteiligten Klassen (und

weitere Dateien, wie z.B. Bilder oder Dokumentationen) und alle Einstellungen wie Bibliotheken, weitere Pfade oder Compiler- und Aufrufparameter und -optionen. Mit dem Öffnen eines Projekts in einer IDE sind alle Projekteinstellungen präsent, mit dem Öffnen eines anderen Projekts dessen Einstellungen.

Eclipse

Wir werden in diesem Semester die IDE "Eclipse" verwenden. Eclipse ist eine "Open Source" Software des Hauses IBM. Für Sie bedeutet das, dass Sie Eclipse kostenlos von der Eclipse-Homepage herunterladen können. Open Source ist auch die Einladung, die Software zu erweitern und zu verbessern. Da Eclipse in Java geschrieben ist, können Sie später evtl. dazu beitragen....
Nach den allgemeinen Einführungen oben sollten Sie sich Eclipse gut selbst erschließen können. Auf der eclipse-Homepage gibt es Einführungen, aber natürlich knnen Sie auch nach deutschen Tutorials oder youtubes googlen, wenn Sie es brauchen.
Sicherlich werden Sie trotzdem immer wieder Fragen haben - fragen Sie Kommilitonen oder Ihren Dozenten, und experimentieren Sie auf eigene Faust!

Hinweis: "JDK" steht für Java Development Kit und ist die ältere Bezeichnung für SDK - Software Development Kit. Zu Hause sollten Sie Eclipse so einstellen, dass dasselbe SDK verwendet wird, das Sie zum Arbeiten von der Kommandozeile benutzen. (Im Labor ist diese Einstellung schon vorgenommen.)
Wenn Sie fertig sind, nehmen Sie Eclipse am besten in Ihr "Repertoire" auf, indem Sie eine Verknüfung auf eclipse.exe im Startmenü speichern.

Weitere Möglichkeiten für Fortgeschrittene:
Wir benutzen Eclipse zur Java-Programmierung. D.h. wir benutzen in Eclipse Werkzeuge zur Bearbeitung von .java-, .class- und .jar-Dateien. Solche Werkzeuge, die in Eclipse integriert sind, heißen "Plugins". Eclipse hat eine offen Plugin-Schnittstelle, so dass jeder eigene Plugins programmieren kann. Das Internet ist voll von kostenlos verfügbaren Eclipse-Plugins, mit denen Sie Ihr Eclipse erweitern können.
Sie können aus Eclipse heraus aber auch die Werkzeuge aufrufen, die in Ihrem Betriebssystem für die entsprechenden Dateitypen registriert sind: Klicken Sie mit der rechten-Maustaste auf eine .java-Datei; im Auswahlfenster erscheint Open with... Hier wird Ihnen u.a. System Editor angeboten - das ist der im Betriebssystem registrierte Java-Editor (vermutlich Textpad). Probieren Sie es aus - es öffnet sich ein Textpad-Fenster.
Wenn Sie in Eclipse eine html-datei anklicken, öffnet sich der Browser - das im Betriebssystem für diesen Dateityp registrierte Werkzeug. Wenn Sie html-Dateien editieren wollen (z.B. für Ihren Laborbericht), benötigen Sie ein anderes Werkzeug. Sie können innerhalb von Eclipse ein beliebigens installiertes Programm diesem Dateityp zuordnen (assoziieren). Wählen Sie Windows>Preferences>Workbench>File Associations. Fügen Sie im oberen Fenster mithilfe des Add-Knopfes *.html hinzu. Im unteren Fenster erscheint die Liste der registrierten Editoren. Wählen Sie External Programs, so erscheint eine andere Liste. Sie können eine davon auswählen oder mit Add einen weiteren hinzufügen (z.B. SelfHtml) und dann selektieren.
Weitere Eclipse-Plugins finden sie im Eclipse-Marketplace. Machen sie es sich in Eclipse „bequem“.

Laborexperimente

,

Experiment 1: Das HalloWelt- und Eliza-Projekt

1. Starten Sie Eclipse und stellen Sie den Workspace auf Ihr Laufwerk, z.B. Z:\eclipse_workspace. Erstellen Sie ein neues Java-Projekt, wählen Sie dabei getrennte Vezeichnisse für Quelle und Kompilat. Nennen Sie es z.B. X1.1.HalloEliza – die Nummerierung verrät schon, dass wir für X1 mehrere Projekte benötigen..
2. Erstellen Sie mit einem Rechtsklick auf ihr Projekt eine „neue Klasse“ HalloWelt und kreuzen sie „static void main“ an (wo landet sie?).
3. Öffnen sie Klasse und tragen sie den Methodenrumpf ein. Machen Sie dabei absichtlich einen

Syntaxfehler – was passiert?
4. Wenn die Fehler behoben Sind, führen Sie die Klasse aus, indem Sei einen Rechtsklick auf die Klasse machen und dann „run as>Java application“ wählen. Was passiert? Wo erscheint die Ausgabe?
5. Versuchen Sie noch andere Befehle zum ausführen.

6. Erstellen sie jetzt eine neue Methode „eliza“,
Signatur: public static void eliza(),
und kopieren Sie Ihren Eliza-Code hinein. Rufen Sie in Ihrer main-Methode jetzt zusätzlich eliza() auf.
7. Tastatur.readln() ist unbekannt. Importieren Sie die Klasse in Ihr Projekt:
Projekt>Rechtsklick>import>from FileSystem. Danach sollte das Projekt wieder auasführbar sein.
(Für Console.readln() s. Experiment 2)

Experiment 2: Bibliothek einbinden und Fehler beheben

1. Das nächste Projekt soll klassischeAlgorithmen heißen.
2. Laden Sie sich Euclid.java herunter und importieren sie die Klasse in Ihr Projekt.
3. Wenn Sie die Klasse öffnen, ist alles voller Fehler!
4. Die Eingabemethode heißt hier Console.readln(), die Klasse Console muss also dem Projekt hinzugefügt werden. Console ist eine von vielen Hilfsklassen, die wir in diesem Semester benötigen. Sie befindet sich in der Bibliothek cs101-lib.jar. Laden sie die Bibliothek herunter und speichern Sie sie an einer gut auffindbaren Stelle, z.B. Z:\utilities.
5. Binden Sie die Bibliothek in die Arbeitsumgebung des Projektes ein, indem sie sie dem „Java Build Path“ hinzufügen (hier sucht der Compiler dann nach Klassen, die im Projekt benutzt werden):
Project>Properties>Java Build Path>Add External Jars und dann die cs101-lib.jar auswählen.
6. Jetzt sollte Console keine Probleme mehr machen. Bearbeiten Sie jetzt die anderen Fehler.
7. Workaround: Wenn Sie mit dem einbinden dr Bibliothek ggroße Probleme haben, können Sie für diesmal auch nur die Klasse Console herunterladen.
8. Wenn alles gut aussieht, dann bringen Sie Euclid zum Laufen. Funktioniert es **Produziert es korrekte Ergebnisse??**

Experiment 3: Debugging

1. Wenn Ihr Programm nicht richtig funktioniert, ist es Zeit für die Käfersuche! Sollte es gar nicht mehr aufhören, können Sie es mit dem roten Knopf über der Ausgabekonsole stoppen.
2. Folgen Sie jetzt den Debugging-Anweisungen in X1a DebuggerIntro (zur Zeit leider nur auf Englisch verfügbar) – finden sie die semantische „Käfer“ und verjagen Sie sie.
3. Wenn‘s gut geklappt hat, dann versuchen sie sich an BinominalCoefficients.java – **freiwillig.**

Experiment 4 – freiwillig, für die Schnellen

1. **Javaeyes.jar ausführen:** Öffnen Sie ein neues Java-Projekt namens JavaEyesJar. Laden Sie javaeyes.jar herunter und binden Sie es als externe Bibliothek en (wie cs101-lib). *(Tipp:javaeyes ist auch in der cs101-lib enthalten)*
2. Führen Sie es über Run>Run as Java Application aus. Was passiert?
3. Selektieren Sie jetzt im Package Explorer links JavaEyes.class in javaeyes.jar. wählen Sie wieder Run>Run as.. wie oben. Jetzt sollte es klappen, denn die Klasse JavaEyes enthält eine main-Methode.

4. **Javaeyes.class ausführen**: Öffnen Sie ein weiteres Projekt mit Namen JavaEyesClass. Wählen Sie aus dem Dateimenü "Import" > Zip File und navigieren Sie wieder zu javaeyes.jar. Der inhalt derJAR-Datei wird dadurch **entpackt** ins Projekt übernommen.
5. Selektieren Sie wieder JavaEyes.class. Wählen Sie Run>Run und ändern Sie das Projekt auf JavaEyesClass. Jetzt müssen Sie erneut javaeyes.JavaEyes als Startklasse angeben. Starten Sie das Programm!

6. **Javaeyes kompilieren und ausführen:** Öffnen Sie das dritte Projekt und nenne Sie es

JavaEyesSource. Wählen Sie gleich im ersten Fenster unter Project layout die Option "create separate source and output folders". Ihr Verzeichnisbaum verzweigt sich dadurch sofort in einen src-Ordner für die Java-Quellen und einen bin-Ordner für die .class-Dateien.

7. Laden Sie die Datei javaeyes-stu.jar* herunter. File>Import und dann "Zip File". Im Nächsten Fenster navigieren Sie zu javaeyes-stu.jar und geben als Zielverzeichnis das src-Verzeichnis Ihres Projekts an (ebenfalls durch Navigieren).
8. Öffenen Sie die Datei JavaEyes.java durch Doppelklick. Sie sehen am Rand einige rote Markierungen, die Sie auf Fehler hinweisen. Wenn Sie mit der Maus auf die Markeirung gehen, erscheint ein Fehlertext. Diesen Fehlertext können Sie auch sehen, wenn Sie im unteren Fenster die Registerkarte "Problems" selektieren. Ein Doppelklick auf eine solche Problemzeile führt Sie an die Fehlerstelle im Code. Könne Sie raten, was die Fehlermeldungen bedeuten? Vergleichens Sie dazu das Projekt mit den beiden vorigen Projekten!
9. Ignorieren Sie die Fehlermarkierungen und kompilieren Sie das Programm mit Project>Build All. Was passiert?
10. Sie haben bestimmt schon erkannt, warum das Programm nicht kompilierbar ist. Haben Sie eine Lösung dafür? Sonst lassen Sie sich von Eclipse helfen: Gehen Sie auf eine der Problemzeilen und drücken Sie die rechte Maustaste. Wählen Sie Quick Fix, dann erhalten Sie Vorschläge, wie Sie das Problem beheben können. Wenn Sie einen Vorschlag auswählen, unternimmt Eclipse die nötigen Schritte. Drücken Sie aber bitte Cancel und machen Sie die nötigen Schritte von Hand - nur heute. (Sie müssen die cs101-Bibliothek ins Projekt einbinden).

11. **JavaEyes verändern:** JavaEyes sind schwarze Augen auf weißem Hintergrund. Wie wäre es mit gelben Augen auf Schwarzem Hintergrund? Suchen Sie die Wörter "black" und "white" und versuchen Sie auf gut Glück, sie zu verändern.
    Zwei Tipps dazu:
    - Lassen Sie Eclipse für Sie suchen: Search>Search
    - Halten Sie sich den Rückweg offen, indem Sie entweder die Originaldatei unter einem anderen Namen speichern (ändern Sie die Endung, z.B. auf ".alt") oder die alte Zeile auskommentieren und dann die veränderte einfügen.
12. Können sie das "Quit" auf dem Knopf noch lesen? Ändern Sie den Text, und versuchenSie, auch die Schriftfarbe oder -größe zu ändern.

Wo finde ich Hilfe?

Hilfe zu Java: Die offizielle Dokumentation zu Ihrer Java-Version, z.B. *http://*docs.oracle.com/javase/8/docs/api sollten Sie immer parat haben - auch wenn Sie jetzt noch wenig verstehen. Sie ist für's Programmieren unerlässlich. Außerdem gibt es eine Reihe guter Tutorials http://docs.oracle.com/javase/tutorial/ , natürlich alles in Englisch...
Hilfe zur cs101-Bibliothek: Laden Sie sich sich die Datei cs101-src.jar herunter, die die Dokumentation enthält.
Hilfe zu Eclipse: Neben der Eclipse-Homepage enthält das Internet eine Fülle von Hilfen und Anleitungen zu Eclipse, auch auf Deutsch, auch als Video: Fragen Sie Google.
Allgemein: Google versteht Sie besser als Sie denken - selbst auf seltsame Compilerfehlermeldungen gibt es viele Antworten zumeist in Foren. Auch hier ist praktisch alles Englisch.
Wenn Sie im Labor sind, fragen Sie ruhig zunächst Kommilitonen, vielleicht haben Sie das Problem gerade schon "geknackt". (Nicht vergessen: Schreiben Sie ins Protokoll, welche Hilfe Sie genutzt haben)

---

*)Links dieser Seite (für die PDF-Version):
Eclipse-Homepage: http://www.eclipse.org/
javaeyes-stu.jar:
http://www.schmiedecke.info/Prg1/Praxis-Downloads/javaeyes-stu.jar
Dokumentation der cs101-Bibliothek:
http://www.schmiedecke.info/Prg1/Praxis-Laboraufgaben/cs101Lib/Doc/index.html
Quelltexte der cs101-Bibliothek:
http://www.schmiedecke.info/Prg1/Praxis-Laboraufgaben/cs101Lib/cs101-src.jar

Hinweis zum Copyright

Viele der Übungen zu diesem Kurs wurden ursprünglich von Prof. Lynn Andrea Stein für Ihren Kurs "IPIJ - Interactive Programming in Java" im Rahmen des Projekts"Rethinking CS101" entwickelt und von Prof. Debora Weber-Wulff teilweise für ihre Kurse weiter bearbeitet.

Die für diesen Kurs angefertigte deutsche Version ist auf Programmieren I für TI an der TFH Berlin abgestimmt und weicht textuell und inhaltlich von den Vorgaben ab.