

Aufgabenblatt

Einführung

Programmierung Praktikum
Prof. Dr. Dirk Eisenbiegler
Hochschule Furtwangen

Aufgabe 1 - Vorbereitung

Diese Aufgabe setzt voraus, dass Sie Eclipse auf Ihrem Computer bereits installiert haben.

A) Starten Sie Eclipse. Innerhalb von Eclipse erscheint ein Begrüßungsfenster. Schließen Sie dieses.

B) Java-Projekt erstellen

Im Menü: *File - New - Java-Project*

In das Feld *Project name* tragen Sie *Programmierung* ein. Drücken Sie auf *finsh*.

Erläuterung: Das Projekt ist ein Verzeichnis auf der Festplatte. In diesem Projekt sollen alle Programme aufbewahrt werden, die in diesem Semester im Programmierpraktikum erstellt werden. Sie benötigen kein weiteres Projekt.

C) Library *prog.jar* einbinden

Die Datei *prog.jar* finden Sie im Intranet und dort im Dateibereich dieser Veranstaltung. Laden Sie diese Datei herunter und kopieren Sie diese mit Copy&Paste innerhalb von Eclipse in das Projektverzeichnis von Programmierung. Markieren Sie die Datei in Eclipse mit der Maus und betätigen Sie die rechte Maustaste. Es erscheint ein Popdown-Menü. Wählen Sie im Popdown-Menü den folgenden Eintrag aus: *Build Path – Add to Build Path*. Jetzt erscheint die Library *prog.jar* im Projektverzeichnis unter *Referenced Libraries*.

Aufgabe 2 - Programmieren mit Eclipse

A) Programm schreiben.

Wählen Sie das Projekt *Programmierung* mit der Maus aus.


Dann im Menü: File - New - Class

In das Feld *name* tragen Sie *ErstesProgramm* ein. Drücken Sie auf *finish*.
Tipp: Statt mit *File-New-Class* können Sie Klassen zukünftig auch mit dem Button  erstellen.


Es entsteht eine Textdatei für den Programmcode. Diese Textdatei enthält bereits das Grundgerüst des Programms. Tippen Sie das folgende Programm ein.

```
public class ErstesProgramm {  
    public static void main(String[] args) {  
        System.out.println("Herzlich willkommen zum Praktikum!");  
        System.out.println("Eine Zufallszahl: " + Math.random());  
        System.out.println("Ende des Programms.");  
    }  
}
```

B) Programm starten



Gehen Sie mit dem Cursor in den Programmtext. Drücken Sie dann mit der Maus neben dem Start-Knopf  auf das kleine schwarze Dreieck, das nach unten zeigt. Sie sehen dann ein Pulldown-Menü. Wählen Sie dort den Punkt *Run As* und dort wiederum *Java Application*. Jetzt wird das Programm übersetzt und gestartet.

C) Programm erneut starten

Wenn Sie ein Programm bereits einmal gestartet haben und es danach noch einmal starten wollen, dann müssen Sie lediglich noch auf den Start-Knopf  drücken (nicht mehr über das kleine schwarze Dreieck rechts daneben). Die Bedeutung des Start-Knopfes: Das zuletzt gestartete Programm noch einmal starten.

D) Fügen Sie die nachfolgende Programmzeile nach der vierten Zeile in Ihr Programm ein. Starten Sie das Programm erneut

```
System.out.println("Gleich ist das Programm zu Ende.");
```

E) Der Programmtext enthält ein Pluszeichen +. Ersetzen Sie dieses durch ein Minuszeichen - und speichern Sie die Datei mit . Eclipse weist Sie auf einen Syntaxfehler hin. Ersetzen Sie das Minuszeichen wieder durch ein Pluszeichen und speichern Sie die Datei erneut mit .

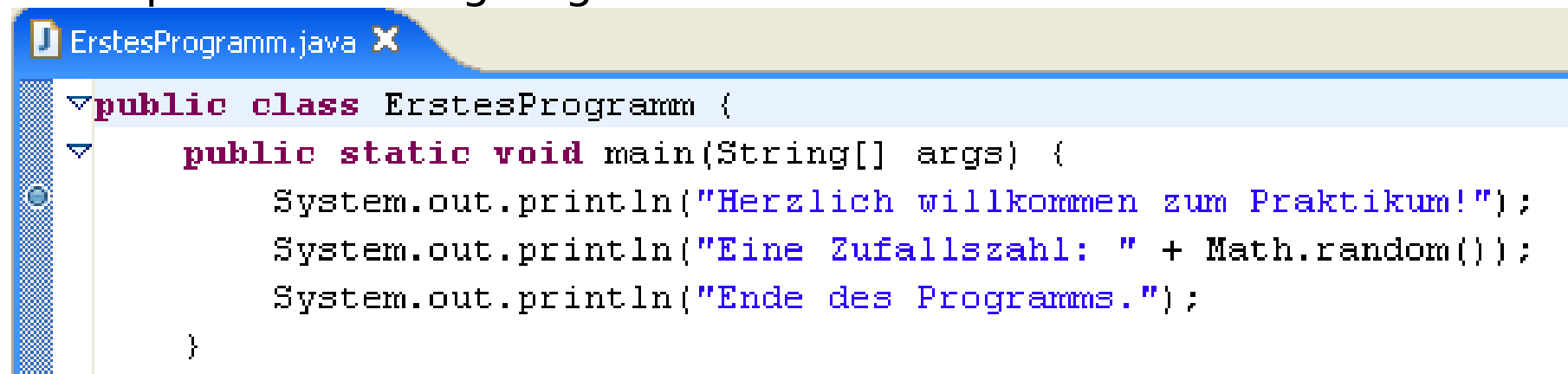
F) Wenn Sie Eclipse verlassen, werden all Ihre Daten gespeichert: Ihr Programmcode und auch alle anderen Einstellungen. Wenn Sie Eclipse wieder starten erscheint der Bildschirm genau so wie Sie ihn zuletzt verlassen haben. Probieren Sie es aus: Eclipse verlassen, Eclipse neu starten.

Aufgabe 3 - Debugging

In dieser Aufgabe soll das Programm von Aufgabe Fehler: Verweis nicht gefunden Schritt für Schritt ausgeführt werden („Debugging“).


A) Breakpoint setzen

Setzen Sie einen Breakpoint vor die dritte Zeile. Ein Breakpoint ist in Eclipse durch eine kleine blaue Kreisfläche dargestellt (siehe Abbildung). Doppelklicken Sie an die entsprechende Stelle und der Breakpoint wird eingefügt.



B) Programm im Debug-Modus starten

Um das Programm zu im Debug-Modus zu starten, gehen Sie wie folgt vor: Gehen Sie mit dem Cursor in den Programmtext. Drücken

Sie dann mit der Maus neben dem Debug-Knopf  auf das kleine schwarze Dreieck, das nach unten zeigt. Sie sehen dann ein Pulldown-Menü. Wählen Sie dort den Punkt *Debug As* und dort wiederum *Java Application*. Jetzt wird das Programm übersetzt und gestartet. Der Programmablauf ist wie ein gewöhnlicher Programmablauf, nur dass das Programm an Breakpoints anhält und von dort aus Schritt für Schritt ausgeführt werden kann. Bei einem gewöhnlichen Start-Vorgang (ohne Debug-Modus) werden Breakpoints ignoriert.

C) Views und Perspectives

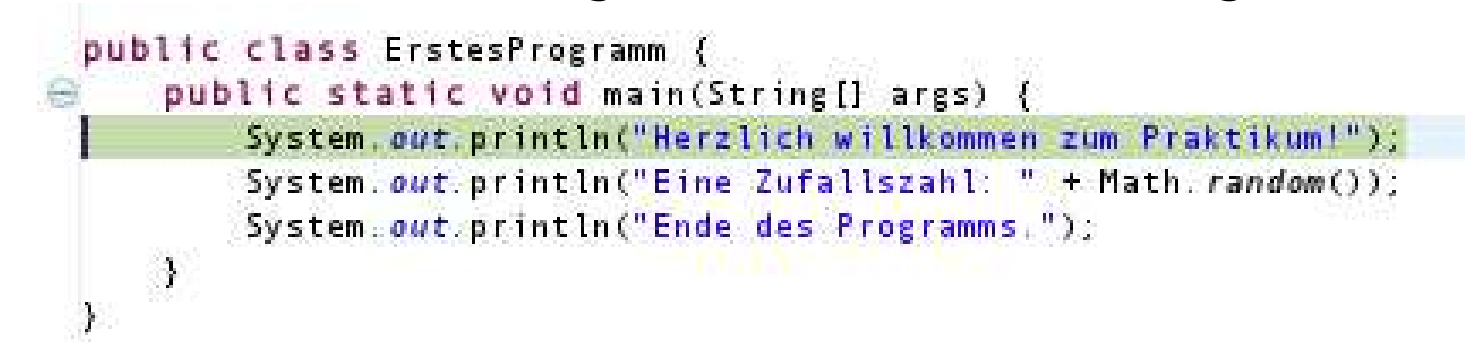
Das Eclipse-Fenster besteht aus mehreren rechteckigen Bereichen (Editoren etc.), die als Views bezeichnet werden. Eine Zusammenstellung mehrerer Views bildet eine Perspective. Rechts

oben im Eclipse-Fenster befinden sich die Knöpfe .




Damit können Sie zwischen der Java-Perspective  und der Debug-Perspective  hin und her-wechseln.

D) Das Programm Schritt für Schritt ausführen

Durch einen Debug-Start wechselt Eclipse automatisch in die Debug-Perspective. Eclipse zeigt Ihnen an, dass das Programm an einer bestimmten Zeile (dem Breakpoint) angehalten wurde. Der Befehl, der als nächstes ausgeführt werden soll, ist grau hinterlegt.




Jetzt können Sie das Programm durch mehrmaliges Drücken des

Knopfes  Befehl für Befehl ausführen. Der Knopf  bewirkt, dass das Programm bis zum nächsten Breakpoint springt. Gibt es im Programm keinen weiteren Breakpoint, so läuft das Programm bis zum Ende. Drücken Sie diesen Knopf, wenn Sie den letzten Befehl erreicht haben. Der Knopf  bewirkt, dass das Programm abgebrochen wird.


E) Ein Programm erneut im Debug-Modus starten

Wenn Sie ein Programm bereits einmal gestartet haben und es danach noch einmal starten wollen, dann müssen Sie lediglich auf

den Debug-Knopf  drücken (nicht mehr über das kleine schwarze Dreieck rechts daneben). Die Bedeutung des Debug-Knopfes: Das zuletzt gestartete Programm noch einmal starten und zwar im Debug-Modus.

Aufgabe 4 - Programm mit Ein- und Ausgabe

A) Legen Sie eine Klasse mit dem Namen *Multiplizierer* an:

- ⇒ Drücken Sie 
- ⇒ Geben Sie als *Source Folder* Programmierung/src an.
- ⇒ Geben Sie als Name *Multiplizierer* an.
- ⇒ Setzen Sie ein Häkchen links von *public static void main(String[] args)*.
- ⇒ Drücken Sie auf Finish.

Unter dem Projekt Programmierung erscheint jetzt im Source-Folder (src) und dort im Default-Package die neue Klasse mit dem Namen *Multiplizierer*. Das Gerüst der Klasse *Multiplizierer* und auch das Gerüst der Methode *main* ist bereits von Eclipse generiert worden.

B) Das Programm soll erstellt werden. Ergänzen Sie die noch fehlenden Stücke im Programmcode.

```
import static prog.ConsoleReader.*;

public class Multiplizierer {
    public static void main(String[] args) {
        int x = readInt("x");
        int y = readInt("y");
        int z = x * y;
        System.out.println(x + " mal " + y + " ergibt " + z);
    }
}
```

C) Starten Sie das Programm. In der Konsole erscheint zunächst x:

Sie werden dadurch aufgefordert, einen Wert für *x* einzugeben. Tippen Sie eine ganze Zahl ein und schließen Sie die Eingabe mit Return ab. In analoger Weise werden Sie anschließend aufgefordert, den Wert von *y* einzugeben. Danach erzeugt das Programm eine Ausgabe.

Aufgabe 5 - Variableninhalte beobachten

Das folgende Programm berechnet alle Zweierpotenzen bis zu einer vorgegebenen Höhe.

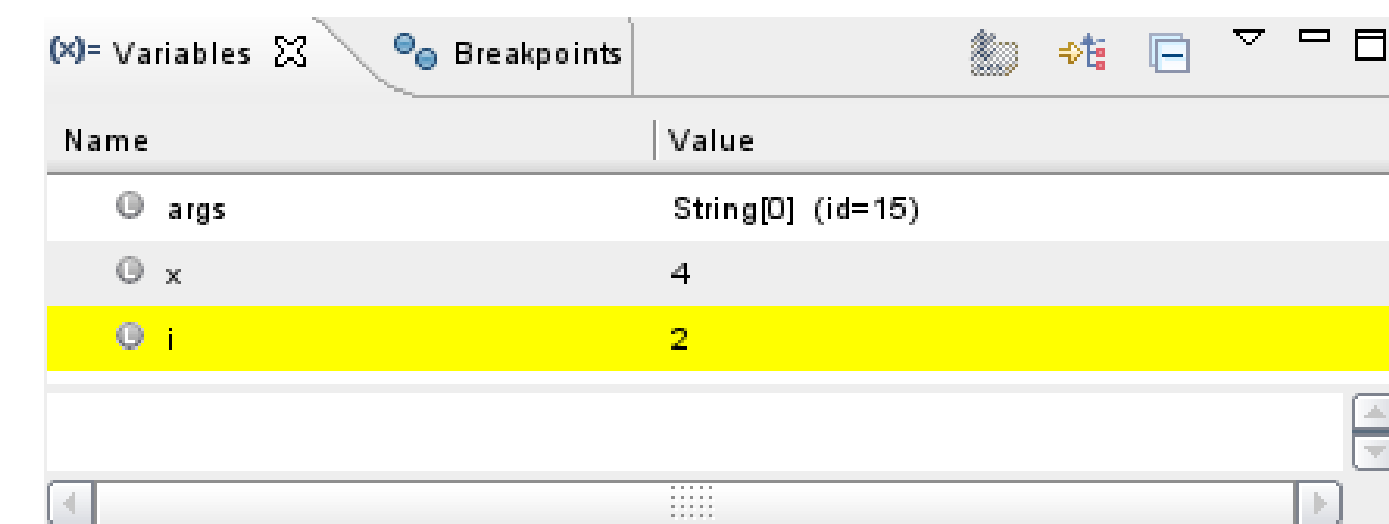
```
public class Zweierpotenzen {
    public static void main(String[] args) {
        int x = 1;
        int i = 0;
        while (x <= 10) {
            System.out.println(i + " : " + x);
            x = x * 2;
            i = i+1;
        }
    }
}
```

A) Erzeugen Sie eine Klasse mit dem Namen *Zweierpotenzen* und übernehmen Sie obigen Programmcode. Starten Sie die Klasse.

B) Setzen Sie vor die Zeile

`int x = 1;`

einen Breakpoint und gehen Sie mit dem Debugger das Programm Schritt für Schritt durch. Verfolgen Sie den Ablauf des Programms und beobachten Sie die Variableninhalte. Die Variablen und ihre Werte werden rechts oben dargestellt. Variablen, die ihren Werte im letzten Schritt gerade geändert haben, werden gelb markiert dargestellt.



Name	Value
args	String[] (id=15)
x	4
i	2